

WotWizard : Mode d'emploi

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

WotWizard construit, à partir de la chaîne de blocs et de la piscine de Duniter, une prédiction des futures entrées des candidats dans la toile de confiance (TdC, ou Wot en anglais) de Duniter. Il utilise pour cela une simulation du mécanisme de Duniter. Lorsque plusieurs dates d'entrées sont possibles pour la même personne, chacune est affectée d'une probabilité. La liste affichée est mise à jour automatiquement toutes les cinq minutes, et tout changement est signalé par une marque visible.

WotWizard contient aussi plusieurs outils complémentaires et un **Explorateur de Toile de Confiance** rapide.

Attention : Il est nécessaire d'utiliser la version 1.7.17 de Duniter ou une version ultérieure de la série 1.7.x.

Comment l'utiliser ?

Ce programme doit être utilisé sur un ordinateur faisant tourner un nœud **Duniter**.

Le nœud **Duniter** doit être configuré pour WotWizard. Pour cela, éditez son fichier de configuration ("conf.json", dans son répertoire ; voir *Configuration* ci-dessous) en mettant la valeur du champ "storage.wotwizard" à "true".

WotWizard est formé d'une partie serveur (**wwServer**) écrite en *Go* (v1.13.6) et tournant sous GNU / Linux. Ce serveur communique par fichiers contenant des requêtes *GraphQL* en entrée et des réponses JSON en sortie. WotWizard possède aussi une partie "interface utilisateur" facultative (**WotWizard.exe**) écrite en *Component Pascal* (BlackBox v1.7.1 à faire tourner sous Wine), qui crée les

requêtes GraphQL par menus et fenêtres interactives et qui affiche les réponses sous forme de textes ou de graphiques.

wwServer doit être lancé dans un répertoire initialement vide. Il crée lui-même les sous-répertoires et fichiers dont il a besoin, le tout restant contenu dans le répertoire initial. La position de la base de données de Duniter, si elle n'est pas à une adresse disque standard, peut lui être indiquée au premier lancement par une option de la ligne de commande.

WotWizard.exe tourne nativement sous Windows. Pour fonctionner avec **wwServer**, il doit être placé dans le même répertoire. Si votre ordinateur tourne sous Linux (nécessaire pour l'instant), installez d'abord "wine".

Mode Serveur

Vous pouvez utiliser WotWizard uniquement comme serveur, si vous voulez assurer vous-même l'affichage de ses résultats, par exemple dans une page web. Après avoir configuré Duniter pour WotWizard (voir ci-dessus), placez l'exécutable **wwServer** dans un répertoire vide. Dans le cas où l'installation de Duniter est standard (sa base de données SQLite doit se trouver dans le fichier "~/.config/duniter/duniter_default/wotwizard-export.db"), il suffit de lancer **wwServer** sans options:

```
$/wwServer
```

Sinon, lors du premier lancement, ajouter l'option -du :

```
$/wwServer -du <chemin vers la base de données de Duniter>
```

Pour les lancements ultérieurs, l'option -du n'est plus utile.

Il est aussi possible de modifier ce chemin en éditant le fichier "rsrc/duniter/init.txt".

La mémoire allouée à WotWizard pour le calcul de ses prévisions est limitée. Sa valeur par défaut est de 430 000 000 octets. Vous pouvez changer cette valeur en éditant le fichier "rsrc/duniter/parameters.txt". Plus elle est grande, plus les prévisions de **wwServer** seront précises, mais si elle est trop grande, **wwServer** peut se bloquer.

Toute modification de "init.txt" ou de "parameters.txt" n'est prise en compte qu'au redémarrage de **wwServer**.

Après une assez courte phase d'initialisation, qui démarre après l'apparition d'un

nouveau bloc dans Dunitier, et que vous pouvez suivre sur le fichier "rsrc/dunitier/log.txt", les fichiers de requête peuvent être déposés dans le sous-répertoire "rsrc/dunitier/Work/Query". Leurs noms n'a pas d'importance et vous pouvez en déposer autant que vous voulez, ils sont immédiatement lus et détruits. Après traitements, les fichiers réponses apparaissent dans le répertoire "rsrc/dunitier/Work/Json" ; voir la règle de correspondance entre les requêtes et les noms des fichiers réponses ci-dessous. En cas d'erreur dans un fichier de requête, des messages explicatifs apparaissent dans log.txt et dans Stderr.

Pour arrêter wwServer, utilisez la frappe de Ctrl-C, ou toute autre méthode standard.

La base de données de WotWizard est contenue dans le fichier "rsrc/dunitier/System/DBase.data" et une nouvelle copie de sauvegarde est créée à chaque lancement; au total deux copies sont conservées : "DBase.data.bak", la plus récente, et "DBase.data.bak1", la précédente ; si vous n'êtes pas sûr de l'intégrité de DBase.data, vous pouvez la remplacer, une fois le logiciel à l'arrêt, en renommant la dernière copie de sauvegarde ou la précédente.

Liste et fonctionnement des requêtes GraphQL

{WWServerStart} : Souscrit un abonnement pour la mise à jour de la fenêtre WotWizard. Ne fait rien si l'abonnement est déjà souscrit (mais peut changer le nom du fichier réponse JSON, voir ci-dessous).

{WWServerStop} : Arrête l'abonnement à la mise à jour de la fenêtre WotWizard. Ne fait rien si l'abonnement n'est pas déjà souscrit.

{WotWizardListFile} : Affiche le fichier WotWizard.

{WotWizardListPerm} : Affiche la liste des permutations de WotWizard.

{IdSearchFind(Hint:"<indice>"){<ensemble de (OldMembers, Members, FutureMembers)>}} : Affiche la liste des identités dont le pseudo ou la clé publique commence par <indice> et dont l'état est "Ancien membre" (OldMembers), "Membre" (Members) ou "Arrivant" (FutureMembers), suivant l'ensemble choisi ; la liste inclut pseudo, hash et état.

{IdSearchFix(Hash:"<hash>"){<ensemble de (Distance, Quality, Centrality)>}} : Affiche les informations relatives à l'identité désignée par <hash> et incluant, ou non, sa distance à la toile de confiance (Distance), sa qualité (Quality) et son degré de contralité (Centrality), suivant l'ensemble choisi.

{History(Uid:"<uid>")} : Affiche l'histoire des entrées / sorties de la toile pour le membre ou l'ancien membre de pseudo <uid>.

{Parameters} : Affiche les paramètres du bloc 0 de la monnaie.

{IdentitiesRevoked} : Affiche la liste des identités révoquées.

{IdentitiesMissing} : Affiche la liste des identités exclues, mais pas encore révoquées.

{IdentitiesMembers} : Affiche la liste des membres actifs.

{CertificationsFrom} : Affiche la liste des certifications en cours de validité, groupées par émetteurs.

{CertificationsTo} : Affiche la liste des certifications en cours de validité, groupées par receveurs.

{Sentries} : Affiche la liste des membres référents.

{Sandbox} : Affiche le contenu de la piscine, trié de plusieurs façons.

{QualitiesDist} : Affiche la distance à la toile de confiance de tous les membres actifs.

{QualitiesQual} : Affiche la qualité de tous les membres actifs.

{CentralitiesAll} : Affiche le degré de centralité de tous les membres actifs.

{MemEnds} : Affiche la date limite de validité de l'adhésion pour tous les membres actifs, dans l'ordre croissant des dates.

{MissEnds} : Affiche la date de révocation pour toutes les identités exclues, mais pas encore révoquées, dans l'ordre croissant des dates.

{CertEnds} : Affiche la date de perte de la cinquième certification pour toutes les identités actives ou exclues, mais pas encore révoquées, dans l'ordre croissant des dates.

{MembersCountAll} : Affiche le nombre de membres, événement par événement (entrées / sorties de la toile).

{MembersCountFluxAll(timeUnit:<unité de temps (s)>)} : Affiche le flux de membres par unité de temps.

{MembersCountFluxPMAll(timeUnit:<unité de temps (s)>)} : Affiche le flux de membres par unité de temps et par membre.

{MembersFirstEntryAll} : Affiche le nombre de premières entrées, événement par événement.

{MembersFEFluxAll(timeUnit:<unité de temps (s)>)} : Affiche le flux de premières entrées par unité de temps.

{MembersFEFluxPMAll(timeUnit:<unité de temps (s)>)} : Affiche le flux de premières entrées par unité de temps et par membre.

{MembersLossAll} : Affiche le nombre de pertes (nombre de membres quittant la toile moins le nombre de ceux qui y reviennent), événement par événement.

{MembersLossFluxAll(timeUnit:<unité de temps (s)>)} : Affiche le flux de pertes par unité de temps.

{MembersLossFluxPMAll(timeUnit:<unité de temps (s)>)} : Affiche le flux de pertes par unité de temps et par membre.

Si un alias est ajouté à une commande (par exemple : "wwResult" dans {wwResult:WWServerStart}), il définit le nom du fichier réponse JSON (c'est-à-dire ici wwResult.json) ; sinon, c'est le nom de la commande elle-même qui est utilisé (ainsi le fichier réponse de {WWServerStart} est WWServerStart.json).

Mode interactif

Ajoutez le fichier "WotWizard.exe" dans le même répertoire que "wwServer".

Lancez WotWizard en double-cliquant sur le fichier "WotWizard.exe", ou avec la commande en ligne, depuis le répertoire choisi (sous Linux) :

```
$wine WotWizard.exe
```

WotWizard.exe ne lance pas wwServer, qu'il faudra lancer indépendamment, avant ou après. Il ne l'arrête pas non plus à sa fermeture.

Après l'ouverture de la fenêtre, choisissez votre langue (et, éventuellement vos

polices de caractères) dans "Édition -> Préférences...".

Vous pouvez alors lancer des commandes avec les menus de l'interface et voir les résultats dans les fenêtres qui s'ouvrent.

La fenêtre WotWizard

Ouvrez la fenêtre WotWizard avec "WotWizard -> Nouvelle fenêtre WotWizard".

Vous pouvez choisir la façon dont la liste est affichée (par noms ou par dates). Sa mise à jour est automatique à chaque apparition d'un nouveau bloc.

Il y a aussi des métadonnées disponibles, analogues au contenu du **Fichier** (voir ci-dessous).

Lorsque la liste change, deux astérisques encadrent le titre, et un bouton "Vérifier" apparaît. En cliquant sur le bouton, on fait disparaître les astérisques et on peut comparer les anciennes et nouvelles listes (par dates et métadonnées) en cliquant sur le bouton "Comparer", ou en utilisant la commande de menu "Édition -> Comparer les textes" (raccourci clavier : F9).

Fichier (*WotWizardListFile*): WotWizard calcule ses prévisions à partir d'un état des lieux, appelé *Fichier*. Il s'agit d'une liste de *dossiers* d'entrées et de *certifications internes*, triée par dates de disponibilité. Chaque dossier contient le pseudo du nouveau venu et la liste des certifications (externes) qu'il a reçues. Les certifications internes sont émises vers des personnes déjà membres. L'affichage, obtenu par la commande de menu "WotWizard -> Fichier", donne les informations suivantes :

- *Certification interne* : une ligne contenant :
 - + le récepteur et l'émetteur de la certification (pseudos) séparés par une flèche (←) ;
 - + la date et l'heure de disponibilité de la certification ;
 - + entre parenthèses, la date et l'heure limite de validité de la certification ;
 - + un résumé de l'état de la certification : OK si elle est disponible et valide, ou KO sinon.
- *Dossier* :
 - + une première ligne décrivant le nouveau venu :
 - * le nombre de certifications principales (celles qui déterminent la date de disponibilité du dossier, voir ci-dessous) ;
 - * son pseudo ;
 - * entre parenthèses :
 - sa date de disponibilité ;
 - sa date de disponibilité minimale, deux mois (sigReplay) après la précédente demande d'adhésion

- * entre parenthèses, la date limite de validité de sa demande d'adhésion ;
 - * le résultat du calcul de la règle de distance, en pourcentage de membres référents accessibles ;
 - * un résumé de l'état du dossier : OK s'il est valide et si les règles de nombres de certifications valides et de distance sont vérifiées, ou KO sinon (la disponibilité des certifications n'est pas prise en compte) ;
- + une ligne supplémentaire par certification externe, identique à celle d'une certification interne, sauf que le récepteur, qui est le nouveau venu, n'est pas mentionné ; ces certifications sont triées par dates de disponibilité.

Certifications principales : Pour qu'un dossier soit valide, il faut qu'il contienne un nombre minimal de certifications ($sigQty = 5$), mais aussi que ces certifications lui permettent de vérifier la règle de distance. Du fait de cette règle de distance, il faut parfois plus que $sigQty$ certifications. Les certifications principales d'un dossier sont celles qui permettent, au minimum, le respect des deux règles. Par extension, si l'une des deux règles n'est pas respectée, on considère toutes les règles du dossier comme principales. Comme les certifications passent dans l'ordre de leurs dates de disponibilité, on considère, dans cet ordre, les $sigQty$ premières règles (ou moins si le dossier est incomplet) ; si elles permettent de respecter la règle de distance, on s'arrête là, sinon on continue avec les certifications suivantes jusqu'à ce que la règle de distance soit satisfaite ou qu'on ait épuisé toutes les certifications du dossier : on a alors trouvé les certifications principales. La date de disponibilité de la dernière certification principale est celle du dossier (il ne sera vraiment disponible à cette date que s'il est valide).

Permutations (*WotWizardListPerm*): La commande "WotWizard -> Permutations" affiche toutes les permutations d'ordre d'entrées prévues par WotWizard avec leurs probabilités. **Attention** : leur nombre peut être très grand et l'affichage très long !

Explorateur de toile de confiance

IdSearchFind & IdSearchFix

Ouvrez l'explorateur avec la commande de menu "Toile de confiance -> chercher une identité".

Voir [la carte de l'explorateur](#).

On peut chercher n'importe quelle identité dans la chaîne de blocs ou dans la piscine en tapant ses premiers caractères ou les premiers caractères de sa clé publique dans le premier champ en haut de la fenêtre et en cliquant sur le bouton "Chercher". Les identités correspondantes s'affichent dans la liste en bas de la fenêtre : choisissez

celle qui vous intéresse. Des informations apparaissent dans le cadre "Identité", ainsi que les certifications reçues et envoyées dans le cadre "Certifications". Il est possible d'afficher l'identité d'un émetteur ou d'un récepteur de certification en cliquant sur bouton "Voir" correspondant ; on peut revenir en arrière avec les boutons fléchés. On peut aussi afficher un texte d'informations sur les certificateurs et certifiés en cliquant sur l'un des boutons "A" (liste des certificateurs ou certifiés actuels, classés par ordres alphabétiques, par date limite des demandes d'adhésion (ou date limite avant révocation si la personne concernée n'est plus membre - "×" devant le nom) et par dates limites de validité des certifications, et liste de tous les certificateurs ou certifiés ayant existé et non-révoqués et pouvant donc à nouveau soit certifier, soit être certifiés). Les noms des certificateurs et certifiés en attente en piscine sont précédés du symbole °.

Les dates affichées sont la date d'enregistrement du membre dans la chaîne de blocs, les dates limites d'expiration, pour l'inscription du membre et pour ses certifications, et la date de disponibilité de la prochaine certification émise dans le champ "Disponibilité" ; si cette dernière est déjà disponible, elle est précédée d'un point d'exclamation (!). Si le membre a demandé l'arrêt de son adhésion, la date limite de son inscription est précédée d'une croix (×). Dans ce cas, il ne peut plus recevoir de certifications, et, sauf renouvellement de son adhésion, il sera exclu à cette date.

Le degré de centralité c d'un membre est le nombre de chemins orientés (certificateur \rightarrow certifié) les plus courts auxquels il appartient est dont la longueur est, au plus, $stepMax$ (5) plus un depuis le certificateur. Le niveau de centralité c' d'un membre est calculé à partir de son degré de centralité c par l'expression :

$$c' = 100 \frac{\ln(1+c)}{\max[\ln(1+c)]}$$

où $\max[\ln(1+c)]$ est le maximum de $\ln(1+c)$ sur l'ensemble des membres, et où \ln est la fonction logarithme népérien.

On peut considérer le *niveau de centralité* d'un membre comme son niveau d'implication dans le respect de la règle de distance des membres qu'il a certifiés, sans préjuger de sa capacité à le faire. Par contre, la *qualité* d'un membre est sa capacité à permettre le respect de la règle de distance par ceux qu'il certifie, sans préjuger de son niveau d'implication.

Un membre qui a une qualité supérieure ou égale à $xpercent$ (= 80%), assure à lui tout seul le respect de la règle de distance par ceux qu'il certifie.

Outils

Paramètres (*Parameters*) : Affiche les paramètres de base de la monnaie.

Identités révoquées (*IdentitiesRevoked*) : Affiche les identités révoquées, avec leurs pseudos et clés publiques, et les dates de leurs adhésions.

Adhésions non-renouvelées (*IdentitiesMissing*) : Affiche les identités dont l'adhésion n'a pas été renouvelée, avec leurs pseudos et clés publiques, les dates de leurs adhésions et la date limite avant révocation.

Identités (*IdentitiesMembers*) : Affiche les identités de tous les membres, avec leurs pseudos et clés publiques, les dates de leurs adhésions et leurs limites de validité.

Certifications depuis... (*CertificationsFrom*) : Affiche toutes les certifications présentes dans la chaîne de blocs, triées par émetteurs, avec leurs dates d'inscription.

Certifications vers... (*CertificationsTo*) : Affiche toutes les certifications présentes dans la chaîne de blocs, triées par récepteurs, avec leurs dates d'inscription.

Membres référents (*Sentries*) : Affiche les identités des membres référents.

Piscine (*Sandbox*) : Affiche les identités et les certifications présentes dans la piscine :

- 1) Identités triées par leurs hashes, avec hash, clé publique, id et date d'expiration
- 2) Identités triées par leurs clés publiques, avec clé publique et hash
- 3) Identités triées par leurs pseudos, avec pseudo et hash
- 4) Certifications triées par les clés publiques de leurs émetteurs, avec clé publique de l'émetteur, hash du récepteur et date d'expiration
- 5) Certifications triées par les clés publiques de leurs récepteurs, avec hash du récepteur, clé publique de l'émetteur et date d'expiration

Distances (*QualitiesDist*) : Affiche les distances de tous les membres à la toile de confiance (proportions des membres référents atteignables en *stepMax* (5) pas au plus), triées par proportions (avec tracé), puis par pseudo.

Qualités (*QualitiesQual*) : Affiche les qualités de tous les membres, triées par qualités (avec tracé), puis par pseudo.

Centralités (*CentralitiesAll*) : Affiche les niveaux de centralité de tous les membres, triés par niveaux de centralité (avec tracé), puis par pseudo.

Limites des adhésions (*MemEnds*) : Affiche les dates limites de validité des adhésions de tous les membres, triées par date.

Limites des adhésions non-renouvelées (*MissEnds*) : Affiche les dates limites

avant révocation de toutes les identités dont l'adhésion n'a pas été renouvelée, triées par date.

Limites des certifications (*CertEnds*) : Affiche les dates limites de validité des adhésions pour manque de certifications de tous les membres, triées par date.

Évolution

Nombre de membres (*MembersCountAll*) : Affiche la liste des dates des blocs où le nombre de membres a changé et les nombres n de membres correspondants depuis le début de la monnaie (avec tracé). Une première liste simplifiée correspond au tracé (avec la date réelle UTC+0, et non la date blockchain). Une seconde liste donne plus d'informations : date réelle, date blockchain, identités entrantes ou sortantes, nombre de membres résultant.

Flux de membres (*MembersCountFluxAll*) : Variation du nombre de membres par unité de temps (ici: le mois). C'est donc la dérivée du nombre n de membres par rapport au temps t : $\frac{dn}{dt}$ (avec tracé).

Flux de membres par membre (*MembersCountFluxPMAll*) : Flux de membres divisé par le nombre n de membres : $\frac{dn}{ndt}$, en pourcentage par unité de temps. C'est aussi la dérivée logarithmique de n par rapport au temps $\left(\frac{d \ln(n)}{dt}\right)$. Lorsque cette valeur reste constante dans le temps, n croît exponentiellement (avec tracé).

Nombre de premières entrées (*MembersFirstEntryAll*) : Affiche la liste des dates des blocs où le nombre de membres entrant pour la première fois dans la toile de confiance a changé et les nombres e de premières entrées correspondantes depuis le début de la monnaie (avec tracé).

Flux de premières entrées (*MembersFEFluxAll*) : Variation du nombre de premières entrées par unité de temps (ici: le mois). C'est donc la dérivée du nombre e de premières entrées par rapport au temps t : $\frac{de}{dt}$ (avec tracé).

Flux de premières entrées par membre (*MembersFEFluxPMAll*) : Flux de premières entrées divisé par le nombre n de membres : $\frac{de}{ndt}$, en pourcentage par unité de temps (avec tracé).

Pertes (*MembersLossAll*) : Affiche la liste des dates des blocs où le nombre de membres a changé et les différences $l=e-n$ correspondantes entre les premières entrées e et les nombres de membres n depuis le début de la monnaie (avec tracé).

Flux de pertes (*MembersLossFluxAll*) : Variation des pertes par unité de temps (ici: le mois). C'est donc la dérivée des pertes l par rapport au temps t : $\frac{dl}{dt}$ (avec tracé).

Flux de pertes par membre (*MembersLossFluxPMAll*) : Flux de pertes divisé par le nombre n de membres : $\frac{dl}{n dt}$, en pourcentage par unité de temps (avec tracé).

Use it and enjoy! - ¡Úsalos y disfrútalos! - Bonne utilisation - Приятного использования - Powodzenia - Viel Spaß

Gérard Meunier