# Content\_update 数据结构整理

## 1. 新增用户授予平台权限

\\libraries\chain\include\graphene\chain\account\_object.hpp

1. **enum** Platform\_Auth\_Permission
2. {
3. Platform\_Permission\_Forward  = 1,   //允许平台转发文章，默认允许
4. Platform\_Permission\_Liked    = 2,   //允许平台点赞或打分，默认允许
5. Platform\_Permission\_Buyout   = 4,   //允许平台买断文章收益，默认允许
6. Platform\_Permission\_Comment  = 8,   //允许平台评论其他文章，默认允许
7. Platform\_Permission\_Reward   = 16,  //允许平台打赏其他文章，默认允许
8. Platform\_Permission\_Transfer = 32   //允许平台动用零钱转账，默认不允许
9. Platform\_Permission\_Post = 64, //允许平台发布文章，默认允许
10. };

更新说明：用户授予平台能代理用户做操作的权限，使用uint32\_t的标志位，方便之后做权限扩展。以上所有平台代理账户操作都需要账户授权，否则操作失效。

## 2. 新增作者授予文章权限

\\libraries\chain\include\graphene\chain\content\_object.hpp

1. **enum** Post\_Permission
2. {
3. Post\_Permission\_Forward = 1,   //文章允许转发，默认允许
4. Post\_Permission\_Liked   = 2,   //文章允许打分、点赞，默认允许
5. Post\_Permission\_Buyout  = 4,   //文章允许出售收益，默认允许
6. Post\_Permission\_Comment = 8,   //文章允许评论，默认允许
7. Post\_Permission\_Reward  = 16   //文章允许打赏，默认允许
8. };

更新说明：作者（外加平台）授予文章的权限，使用uint32\_t的标志位。所有对文章的相关操作，需要作者打开文章相关权限，否则操作实效。

## 3. 新增文章类型

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **enum** Post\_Type
2. {
3. Post\_Type\_Post = 0,   //新文章
4. Post\_Type\_Comment = 1,   //评论
5. Post\_Type\_forward = 2,   //转发的文章
6. Post\_Type\_forward\_And\_Modify = 3,  //转发且修改的文章
8. Post\_Type\_Default
9. };

更新说明：添加了文章的类型，具体见上。

## 4. 扩展account\_auth\_platform\_operation

\\libraries\chain\include\graphene\chain\protocol\account.hpp

1. **struct** account\_auth\_platform\_operation : **public** base\_operation
2. {
3. **struct** ext
4. {
5. optional<share\_type> limit\_for\_platform;  //扩展： 账户授予平台使用零钱的最大限额。当限额使用完了，所有平台动用账户零钱的操作将失败，直到账户提高平台限额。
6. optional<uint32\_t>   permission\_flags;   //扩展： 账户授予平台权限，权限按标志位设定，具体设定参见 Platform\_Auth\_Permission
7. };
9. **typedef** static\_variant< ext > extension\_parameter;
11. **struct** fee\_parameters\_type
12. {
13. uint64\_t fee              = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
14. uint64\_t min\_real\_fee     = 0;
15. uint16\_t min\_rf\_percent   = 0;
16. extensions\_type   extensions;
17. };
19. fee\_type                   fee;
20. account\_uid\_type           uid;
21. account\_uid\_type           platform;
23. optional< flat\_set<extension\_parameter> > extensions;
24. };

更新说明：账户授权平台，在原来的account\_auth\_platform\_operation基础上，扩展添加了对授权平台动用零钱的限额，以及平台各类代理操作的限定。

## 5. 扩展post\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **class** Recerptor\_Parameter  //扩展：文章受益人参数
2. {
3. **public**:
4. uint16\_t       cur\_ratio;   //扩展: 当前的文章收益占比
5. **bool**           to\_buyout;   //扩展：收益是否出售，出售比例不得大于自己的当前占比，当该文章目前的受益人（包括平台）已经满了5人，必须全部出售自己的收益占比
6. uint16\_t       buyout\_ratio;  //扩展：收益出售比例，
7. share\_type     buyout\_price;  //扩展：收益出售价格
8. time\_point\_sec buyout\_expiration = time\_point\_sec::maximum();  //扩展：收益出售过期时间，当时间过期，该收益出售将无效
9. }
11. **struct** post\_operation : **public** base\_operation
12. {
13. **struct** ext
14. {
15. uint8\_t                                                post\_type = Post\_Type\_Post;   //扩展：文章类型，参见Post\_Type
16. optional<share\_type>                                   forward\_price;   //扩展：转发价
17. optional< map<account\_uid\_type, Recerptor\_Parameter> > receiptors;   //扩展：文章受益人列表，列表强制包括文章平台，且强制占比25%，不可修改，包括文章作者，至少占比25%。收益列表受益人数必须是[2,5]之间
18. optional<license\_lid\_type>                             license\_lid;   //扩展：文章许可id
19. uint32\_t                                               permission\_flags = 0xFFFFFFFF;  //扩展：文章权限，参见Post\_Permission
20. };
21. **typedef** static\_variant< ext > extension\_parameter;
23. **struct** fee\_parameters\_type {
24. uint64\_t fee              = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
25. uint32\_t price\_per\_kbyte  = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
26. uint64\_t min\_real\_fee     = 0;
27. uint16\_t min\_rf\_percent   = 0;
28. extensions\_type   extensions;
29. };
31. fee\_type                     fee;
33. post\_pid\_type                post\_pid;
34. account\_uid\_type             platform = 0;
35. account\_uid\_type             poster = 0;
36. optional<account\_uid\_type>   origin\_poster;
37. optional<post\_pid\_type>      origin\_post\_pid;
38. optional<account\_uid\_type>   origin\_platform;
40. string                       hash\_value;
41. string                       extra\_data = "{}";
42. string                       title;
43. string                       body;
45. optional< flat\_set<extension\_parameter> > extensions;
46. };

更新说明：在原来的post\_operation基础上，添加了文章类型，文章转发价，文章许可，文章权限，以及文章受益人列表等扩展属性。

## 6. 扩展post\_update\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **struct** post\_update\_operation : **public** base\_operation
2. {
3. **struct** ext
4. {
5. optional<share\_type>           forward\_price; //扩展：修改的文章转发价
6. optional<account\_uid\_type>     receiptor; //扩展：受益人账户id
7. optional<**bool**>                 to\_buyout; //扩展：是否出售收益
8. optional<uint16\_t>             buyout\_ratio; //扩展：出售收益比例，出售比例不得大于自己的当前占比。当该文章目前的受益人（包括平台）已经满了5人，必须全部出售自己的收益占比。
9. optional<share\_type>           buyout\_price; //扩展：出售收益价格
10. optional<time\_point\_sec>       buyout\_expiration; //扩展：出售挂单过期时间
11. optional<license\_lid\_type>     license\_lid; //扩展：文章许可id
12. optional<uint32\_t>             permission\_flags; //扩展：文章权限，参见Post\_Permission
13. };
14. **typedef** static\_variant< ext > extension\_parameter;
16. **struct** fee\_parameters\_type {
17. uint64\_t fee              = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
18. uint32\_t price\_per\_kbyte  = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
19. uint64\_t min\_real\_fee     = 0;
20. uint16\_t min\_rf\_percent   = 0;
21. extensions\_type   extensions;
22. };
24. fee\_type                     fee;
26. account\_uid\_type             platform;
27. account\_uid\_type             poster;
28. post\_pid\_type                post\_pid;
30. optional< string >           hash\_value;
31. optional< string >           extra\_data;
32. optional< string >           title;
33. optional< string >           body;
35. optional< flat\_set<extension\_parameter> > extensions;
36. };

更新说明：在原来的post\_update\_operation基础上，添加了文章转发价，文章许可id，文章权限，以及文章受益人的是否出售文章受益，出售收益比，出售价格，挂单有效时间等扩展属性。

## 7. 新增score\_create\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **struct** score\_create\_operation : **public** base\_operation
2. {
3. **struct** fee\_parameters\_type {
4. uint64\_t fee = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
5. uint32\_t price\_per\_kbyte = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
6. uint64\_t min\_real\_fee = 0;
7. uint16\_t min\_rf\_percent = 0;
8. extensions\_type   extensions;
9. };
11. fee\_type                     fee;  //手续费
13. account\_uid\_type             from\_account\_uid;  //打分或点赞发起人账户id
14. account\_uid\_type             platform;   //文章所在平台
15. account\_uid\_type             poster;   //文章的作者
16. post\_pid\_type                post\_pid;   //文章的id
17. int8\_t                       score;   //文章打的分数，分数范围为-5~5
18. share\_type                   csaf;   //打分或点赞花费的积分
20. extensions\_type              extensions;  //扩展项
21. };

更新说明：新增score\_create\_operation，用于账户对文章进行打分或者点赞。文章需要开放打分权限，用户需要授权平台打分权限，否则操作失败。数据库底层新建score\_object来存储打分信息。每一个分配周期会参考打分信息来评定收益。超过最大的保存时间（由理事会参数决定）的，score\_object会被数据库删除。

## 8. 新增reward\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **struct** reward\_operation : **public** base\_operation
2. {
3. **struct** fee\_parameters\_type {
4. uint64\_t fee = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
5. uint32\_t price\_per\_kbyte = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
6. uint64\_t min\_real\_fee = 0;
7. uint16\_t min\_rf\_percent = 0;
8. extensions\_type   extensions;
9. };
11. fee\_type                     fee;  //手续费
13. account\_uid\_type             from\_account\_uid; //打赏发起人账户id
14. account\_uid\_type             platform;   //文章所在平台
15. account\_uid\_type             poster;   //文章作者
16. post\_pid\_type                post\_pid;  //文章id
17. asset                        amount;   //打赏给文章的资产，可以是任意的资产
19. extensions\_type              extensions;  //扩展项
20. };

更新说明：新增reward\_operation，用于账户对文章进行打赏。文章需要开放打赏权限，否则失败。打赏的资产可以是任意链上资产（不限定为YOYO），打赏的资产由文章的受益人门按照收益比例平分，该收益直接结算，不进入收益分配环节。

## 9. 新增reward\_proxy\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **struct** reward\_proxy\_operation : **public** base\_operation
2. {
3. **struct** fee\_parameters\_type {
4. uint64\_t fee = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
5. uint32\_t price\_per\_kbyte = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
6. uint64\_t min\_real\_fee = 0;
7. uint16\_t min\_rf\_percent = 0;
8. extensions\_type   extensions;
9. };
11. fee\_type                     fee;  //手续费
13. account\_uid\_type             from\_account\_uid; //代理打赏的账户id
14. account\_uid\_type             platform;   //文章所在平台
15. account\_uid\_type             poster;   //文章作者
16. post\_pid\_type                post\_pid;  //文章的id
17. share\_type                   amount;   //打上给文章的金额，代理打赏是平台代理用户操作，此处是动用账户的零钱，会使用账户授予平台的零钱限额
19. extensions\_type              extensions;  //扩展项
20. };

更新说明：新增reward\_proxy\_operation，用于平台代理账户来打赏文章。文章需要开放打赏权限，用户需要授权平台打赏权限，否则失败。该操作动用账户授予平台的零钱，消耗账户授予平台的额度，因此资产必须是YOYO，且平台必须是账户允许代理打赏的。打赏的YOYO由文章的受益人门按照收益比例平分，该收益直接结算，不进入收益分配环节。

## 10. 新增buyout\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **struct** buyout\_operation : **public** base\_operation
2. {
3. **struct** fee\_parameters\_type {
4. uint64\_t fee = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
5. uint32\_t price\_per\_kbyte = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
6. uint64\_t min\_real\_fee = 0;
7. uint16\_t min\_rf\_percent = 0;
8. extensions\_type   extensions;
9. };
11. fee\_type                     fee;  //手续费
13. account\_uid\_type             from\_account\_uid; //买断文章受益的发起账户id
14. account\_uid\_type             platform;   //文章所在平台
15. account\_uid\_type             poster;   //文章的作者
16. post\_pid\_type                post\_pid;  //文章的id
17. account\_uid\_type             receiptor\_account\_uid; //要买断的受益人的id
19. extensions\_type              extensions;  //扩展项
20. };

更新说明：新增buyout\_operation，用于文章收益人对文章收益比的交易。文章需要开发买断权限，用户需要授权平台买断权限，否则失败。该操作动用账户零钱，消耗用户授予平台的额度。用于买断的YOYO会从发起账户的零钱转给收益卖出账户的零钱。

## 11. 新增license\_create\_operation

\\libraries\chain\include\graphene\chain\protocol\content.hpp

1. **struct** license\_create\_operation : **public** base\_operation
2. {
3. **struct** fee\_parameters\_type {
4. uint64\_t fee = 1 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
5. uint32\_t price\_per\_kbyte = 10 \* GRAPHENE\_BLOCKCHAIN\_PRECISION;
6. uint64\_t min\_real\_fee = 0;
7. uint16\_t min\_rf\_percent = 0;
8. extensions\_type   extensions;
9. };
11. fee\_type                     fee;  //手续费
13. license\_lid\_type             license\_lid;   //许可的id
14. account\_uid\_type             platform = 0;   //许可所在平台
15. uint8\_t                      type;   //许可类型，目前范围未定
16. string                       hash\_value;   //许可的hash
17. string                       extra\_data = "{}"; //许可的额外内容
18. string                       title;   //许可的标题
19. string                       body;   //许可的内容
21. extensions\_type              extensions;  //扩展项
22. };

更新说明：新增license\_create\_operation，用于平台用户新建文章许可。数据库底层新建license\_object来存储。许可在链上只可以新建，不可以修改。当文章许可声明需要修改的时候，需新建许可内容，并更新文章许可号。

## 12. 新增score\_object

\\libraries\chain\include\graphene\chain\content\_object.hpp

1. **class** score\_object : **public** graphene::db::abstract\_object<score\_object>
2. {
3. **public**:
4. **static** **const** uint8\_t space\_id = implementation\_ids;
5. **static** **const** uint8\_t type\_id = impl\_score\_object\_type;
7. account\_uid\_type    from\_account\_uid; //打分或点赞的发起账户id
8. account\_uid\_type    platform;         //文章所在平台
9. account\_uid\_type    poster;           //文章作者
10. post\_pid\_type       post\_pid;         //文章的id
11. int8\_t              score;            //打分的分数，范围为-5~5
12. share\_type          csaf;             //花费的积分数
13. uint64\_t            period\_sequence;  //周期号
14. share\_type          profits;          //点赞收益
15. time\_point\_sec      create\_time;      //创建时间
16. };

更新说明：数据库新增score\_object用于存储点赞打分信息，每一个分配周期会参考打分信息来评定收益。超过最大的保存时间（由理事会参数决定）的，score\_object会被数据库删除。

## 13. 新增license\_object

\\libraries\chain\include\graphene\chain\content\_object.hpp

1. **class** license\_object : **public** graphene::db::abstract\_object<license\_object>
2. {
3. **public**:
4. **static** **const** uint8\_t space\_id = implementation\_ids;
5. **static** **const** uint8\_t type\_id = impl\_license\_object\_type;
7. license\_lid\_type             license\_lid;  //许可的id
8. account\_uid\_type             platform;     //平台账户的id
9. uint8\_t                      license\_type; //许可类型，暂时没有预定具体类型
11. string                       hash\_value;   //许可的hash
12. string                       extra\_data;   //许可的额外内容
13. string                       title;   //许可的标题
14. string                       body;   //许可的内容
16. time\_point\_sec               create\_time;   //许可的创建时间
17. };

更新说明：数据库新增license\_object用于存储平台用户新建的许可信息。许可只能是平台来创建，且不可被修改。用户发布文章的时候，需要选择平台发布的相应的许可。

## 14. 新增active\_post\_object

\\libraries\chain\include\graphene\chain\content\_object.hpp

1. **class** active\_post\_object : **public** graphene::db::abstract\_object < active\_post\_object >
2. {
3. **public**:
4. **static** **const** uint8\_t space\_id   = protocol\_ids;
5. **static** **const** uint8\_t type\_id    = active\_post\_object\_type;
7. **struct** receiptor\_detail //收益人的收益详细
8. {
9. share\_type  forward;                          //转发收益和
10. share\_type  post\_award;                       //来自内容资金池的收益
11. flat\_map<asset\_aid\_type, share\_type> rewards; //打赏收益和
12. };
14. account\_uid\_type                       platform;   // 平台账号
15. account\_uid\_type                       poster;   // 作者账号
16. post\_pid\_type                          post\_pid;   // 同一账号同一平台下帖子唯一标识（pid）
17. vector<score\_id\_type>                  scores;   // 打分标识集合
18. share\_type                             total\_csaf; // 文章总共积分
19. flat\_map<asset\_aid\_type, share\_type>   total\_rewards;   // 文章打赏
20. uint64\_t                               period\_sequence;   // 文章所属奖励周期数
22. **bool**                                   positive\_win;            //true: 正方胜(赞大于踩) false:反方胜（踩大于赞）
23. share\_type                             post\_award;              //文章的来自平台资金池(内容)的收益
24. share\_type                             forward\_award;           //文章的打赏收益和
25. flat\_map<account\_uid\_type, receiptor\_detail> receiptor\_details; //收益人的收益表
26. }

更新说明：文章在获得点赞或打赏时创建此object（如同一评奖周期内已存在该object，修改属性即可），用于内容评奖周期到来时计算评奖所用。

数据库默认保存最近十个评奖周期内的object，新增命令行参数active-post-periods，用于设置数据库可保存多少期object。

## 15. 理事会参数扩展

1. **struct** content\_parameter\_extension\_type
2. {
3. uint32\_t   content\_award\_interval              = GRAPHENE\_DEFAULT\_CONTENT\_AWARD\_INTERVAL;                // 内容奖励周期，评奖周期。初始值为0，  0代表关闭，单位是秒。
4. uint32\_t   platform\_award\_interval             = GRAPHENE\_DEFAULT\_PLATFORM\_AWARD\_INTERVAL;               // 平台投票奖励周期。初始值为0 ，0代表关闭，单位是秒。
5. share\_type max\_csaf\_per\_approval               = GRAPHENE\_DEFAULT\_MAX\_CSAF\_PER\_APPROVAL;                 // 单次点赞消耗最大积分。初始值为1000积分 ，需要注意积分精度。
6. uint32\_t   approval\_expiration                 = GRAPHENE\_DEFAULT\_APPROVAL\_EXPIRATION;                   // 点赞过期时间, 同一篇文章多久可以点赞一次。初始值为1年  ，单位是秒，
7. share\_type min\_effective\_csaf                  = GRAPHENE\_DEFAULT\_MIN\_EFFECTIVE\_CSAF;                    // 文章参与评奖所需最小积分。初始值为100积分 ，注意积分精度
8. share\_type total\_content\_award\_amount          = GRAPHENE\_DEFAULT\_TOTAL\_CONTENT\_AWARD\_AMOUNT;            // 【内容资金池】总共文章奖励金额。初始值为0，一年量,每年给文章的总奖励。要考虑精度。
9. share\_type total\_platform\_content\_award\_amount = GRAPHENE\_DEFAULT\_TOTAL\_PLATFORM\_CONTENT\_AWARD\_AMOUNT;   // 【平台资金池（内容）】平台从文章中收益奖励金额。初始值为0，一年量  。
10. share\_type total\_platform\_voted\_award\_amount   = GRAPHENE\_DEFAULT\_TOTAL\_PLATFORM\_VOTED\_AWARD\_AMOUNT;     // 【平台资金池（投票）】总共平台投票奖励金额。初始值为0，一年量。平台投票激励
11. share\_type platform\_award\_min\_votes            = GRAPHENE\_DEFAULT\_PLATFORM\_AWARD\_MIN\_VOTES;              // 平台参与评奖所需最小投票数。初始值为10  ，需考虑精度
12. uint32\_t   platform\_award\_requested\_rank       = GRAPHENE\_DEFAULT\_PLATFORM\_AWARD\_REQUESTED\_RANK;         // 平台参与评奖所需投票排名数。初始值为100  ，即：前100名才有激励资格。
13. uint32\_t   platform\_award\_basic\_rate           = GRAPHENE\_DEFAULT\_PLATFORM\_AWARD\_BASIC\_RATE;             // 平台投票奖励中基础比例（平分给前100名平台）。初始值为20%
14. uint32\_t   casf\_modulus                        = GRAPHENE\_DEFAULT\_CASF\_MODULUS;                          // 积分绝对值+ （评分\*积分/5）\*系数；加权平均系数。初始值为200%
15. uint32\_t   post\_award\_expiration               = GRAPHENE\_DEFAULT\_POST\_AWARD\_EXPIRATION;                 // 文章评奖过期时间。初始值为1年  ，超过这个时间丧失评奖资格。单位是秒
16. uint32\_t   approval\_casf\_min\_weight            = GRAPHENE\_DEFAULT\_APPROVAL\_MIN\_CASF\_WEIGHT;              // 点赞积分最小权重。初始值为50%
17. uint32\_t   approval\_casf\_first\_rate            = GRAPHENE\_DEFAULT\_APPROVAL\_CASF\_FIRST\_RATE;              // 计算有效积分比例一。初始值为30%
18. uint32\_t   approval\_casf\_second\_rate           = GRAPHENE\_DEFAULT\_APPROVAL\_CASF\_SECOND\_RATE;             // 计算有效积分比例二。初始值为70%//有效积分比例二应该大于有效积分比例一
19. uint32\_t   receiptor\_award\_modulus             = GRAPHENE\_DEFAULT\_RECEIPTOR\_AWARD\_MODULUS;               // 反方胜时（踩大于赞）文章收益人奖励系数。初始值为80%
20. uint32\_t   disapprove\_award\_modulus            = GRAPHENE\_DEFAULT\_DISAPPROVE\_AWARD\_MODULUS;              // 反方胜时（踩大于赞）点踩者奖励系数。初始值为120%
21. };

## 16. 扩展platform\_object

1. **class** platform\_object : **public** graphene::db::abstract\_object<platform\_object>
2. {
3. **public**:
4. **static** **const** uint8\_t space\_id = protocol\_ids;
5. **static** **const** uint8\_t type\_id  = platform\_object\_type;
7. **struct** Platform\_Period\_Profits   //平台每周期详细收益
8. {
9. flat\_map<asset\_aid\_type, share\_type>   rewards\_profits;       //每周期文章打赏的收益和
10. share\_type                             foward\_profits    = 0; //每周期文章转发的收益和
11. share\_type                             post\_profits      = 0; //每周期来自内容资金池的收益
12. share\_type                             platform\_profits  = 0; //每周期来自平台资金池(内容)的收益
13. };
15. account\_uid\_type owner = 0;
16. string name;
17. uint32\_t sequence;
19. **bool** is\_valid = **true**;
20. uint64\_t total\_votes = 0;
21. string url;
23. uint64\_t pledge = 0;
24. fc::time\_point\_sec  pledge\_last\_update;
25. uint64\_t            average\_pledge = 0;
26. fc::time\_point\_sec  average\_pledge\_last\_update;
27. uint32\_t            average\_pledge\_next\_update\_block;
29. map<time\_point\_sec, share\_type>        vote\_profits;   //平台投票收益,来自平台资金池(投票) ,map最大size为命令行参数active-post-periods 所设置值，默认为10
30. map<uint32\_t, Platform\_Period\_Profits> period\_profits; //平台每个周期所有的收益， ,map最大size为命令行参数active-post-periods 所设置值，默认为10
32. string extra\_data = "{}";
34. time\_point\_sec create\_time;
35. time\_point\_sec last\_update\_time;
36. };

更新说明：在platform\_object里扩展了平台的投票收益表，以及平台每个周期的其他收益表。详见注释。

## 17. 扩展dynamic\_global\_property\_object

1. **class** dynamic\_global\_property\_object : **public** abstract\_object<dynamic\_global\_property\_object>
2. {
3. **public**:
4. **static** **const** uint8\_t space\_id = implementation\_ids;
5. **static** **const** uint8\_t type\_id  = impl\_dynamic\_global\_property\_object\_type;
7. uint32\_t                        head\_block\_number = 0;
8. block\_id\_type                   head\_block\_id;
9. time\_point\_sec                  time;
10. account\_uid\_type                current\_witness = 0;
11. time\_point\_sec                  next\_maintenance\_time;
12. time\_point\_sec                  last\_budget\_time;
13. share\_type                      witness\_budget;
14. share\_type                      budget\_pool;
15. share\_type                      total\_budget\_per\_block = 0;
16. uint32\_t                        next\_budget\_adjust\_block = 0;
17. uint32\_t                        next\_committee\_update\_block = 0;
18. committee\_proposal\_number\_type  next\_committee\_proposal\_number = 1;
19. uint32\_t                        accounts\_registered\_this\_interval = 0;
21. time\_point\_sec                  next\_content\_award\_time = time\_point\_sec(0);   //下一次内容奖励时间，默认为0；内容奖励开启时设置该时间，关闭时，设置为0
22. time\_point\_sec                  last\_content\_award\_time = time\_point\_sec(0);   //上一次内容奖励时间，默认为0；内容奖励开启时设置该时间，关闭时，设置为0
23. time\_point\_sec                  next\_platform\_voted\_award\_time = time\_point\_sec(0);   //下一次平台投票奖励时间，默认为0；平台投票奖励开启时设置该时间，关闭时，设置为0
24. time\_point\_sec                  last\_platform\_voted\_award\_time = time\_point\_sec(0);   //上一次平台投票奖励时间，默认为0；平台投票奖励开启时设置该时间，关闭时，设置为0
25. uint64\_t                        current\_active\_post\_sequence = 0;   //当前的评奖周期的编号即当前活跃文章周期数
27. **bool**                            content\_award\_enable = **false**;   //内容评奖开关，内容奖励开启时设置为true，关闭时设置为false;用于限制active\_post\_object的创建与修改，即当content\_award\_enable为false时，不能创建与修改active\_post\_object,反之即可
28. /\*\*
29. \*  Every time a block is missed this increases by
30. \*  RECENTLY\_MISSED\_COUNT\_INCREMENT,
31. \*  every time a block is found it decreases by
32. \*  RECENTLY\_MISSED\_COUNT\_DECREMENT.  It is
33. \*  never less than 0.
34. \*
35. \*  If the recently\_missed\_count hits 2\*UNDO\_HISTORY then no new blocks may be pushed.
36. \*/
37. uint32\_t          recently\_missed\_count = 0;
39. /\*\*
40. \* The current absolute slot number.  Equal to the total
41. \* number of slots since genesis.  Also equal to the total
42. \* number of missed slots plus head\_block\_number.
43. \*/
44. uint64\_t                current\_aslot = 0;
46. /\*\*
47. \* used to compute witness participation.
48. \*/
49. fc::uint128\_t recent\_slots\_filled;
51. /\*\*
52. \* dynamic\_flags specifies chain state properties that can be
53. \* expressed in one bit.
54. \*/
55. uint32\_t dynamic\_flags = 0;
57. uint32\_t last\_irreversible\_block\_num = 0;
59. **enum** dynamic\_flag\_bits
60. {
61. /\*\*
62. \* If maintenance\_flag is set, then the head block is a
63. \* maintenance block.  This means
64. \* get\_time\_slot(1) - head\_block\_time() will have a gap
65. \* due to maintenance duration.
66. \*
67. \* This flag answers the question, "Was maintenance
68. \* performed in the last call to apply\_block()?"
69. \*/
70. maintenance\_flag = 0x01
71. };
72. };
73. }}

# Database\_api 接口修改整理

## 1. 新增获取点赞打分

1. score\_object get\_score(**const** account\_uid\_type platform,     //文章所在平台
2. **const** account\_uid\_type poster\_uid,   //文章作者
3. **const** post\_pid\_type    post\_pid,     //文章id
4. **const** account\_uid\_type from\_account  //点赞打分人的账户id
5. )**const**;

返回 score\_object的详细信息。

示例：*get\_score platform\_account1, test\_account1, 1, test\_account2*

1. vector<score\_object> list\_scores(**const** account\_uid\_type platform,              //文章所在平台
2. **const** account\_uid\_type poster\_uid,            //文章作者
3. **const** post\_pid\_type    post\_pid,              //文章id
4. **const** uint32\_t         limit,                 //展示数目限制
5. **const** **bool**             list\_cur\_period = **true** //是否只展示当前收益分配周期 true:只展示当前收益周期的点赞；false：展示所有点赞。
6. )**const**;

返回该文章所有的点赞信息。

示例：*list\_scores platform\_account1, test\_account1, 1, 100, true*

## 2. 新增获取许可信息

1. license\_object get\_license(**const** account\_uid\_type platform,    //平台的账户id
2. **const** license\_lid\_type license\_lid  //许可的id
3. )**const**;

返回license\_object的详细信息

示例：*get\_license platform\_account1, 1*

1. vector<license\_object> list\_licenses(**const** account\_uid\_type platform, //平台账户的id或者名字
2. **const** uint32\_t limit             //展示数目限制
3. )**const**;

返回该平台下所有的许可信息。

示例：*list\_licenses platform\_account1, 100*

## 3. 新增获取文章收益

1. vector<active\_post\_object> get\_post\_profits\_detail(**const** uint32\_t         begin\_period, //起始周期号
2. **const** uint32\_t         end\_period,   //截至周期号
3. **const** account\_uid\_type platform,     //文章所在平台
4. **const** account\_uid\_type poster,       //文章的作者
5. **const** post\_pid\_type    post\_pid      //文章的id
6. )**const**;

返回从起始周期到截至周期的文章收益详情vector<active\_post\_object>。

示例：*get\_post\_profits\_detail 1 5 platform\_account1 auther\_account 1*

## 4. 新增获取平台收益

1. **struct** Post\_Object\_Index  //文章索引
2. {
3. account\_uid\_type platform; //文章所在平台
4. account\_uid\_type poster;   //文章作者
5. post\_pid\_type    postid;   //文章id
6. };
7. **struct** Platform\_Period\_Profit\_Detail  //平台周期收益详情
8. {
9. uint32\_t                               cur\_period;       //当前周期号
10. account\_uid\_type                       platform\_account; //平台所有人账户
11. string                                 platform\_name;    //平台名称
13. flat\_map<asset\_aid\_type, share\_type>   rewards\_profits;      //每周期文章打赏的收益和
14. share\_type                             foward\_profits = 0;   //每周期文章转发的收益和
15. share\_type                             post\_profits = 0;     //每周期来自内容资金池的收益
16. share\_type                             platform\_profits = 0; //每周期来自平台资金池(内容)的收益
18. vector< Post\_Object\_Index >            active\_post\_pids;     //所有有收益的文章id表
19. };
21. vector<Platform\_Period\_Profit\_Detail> get\_platform\_profits\_detail(**const** uint32\_t         begin\_period, //起始周期号
22. **const** uint32\_t         end\_period,   //截至周期号
23. **const** account\_uid\_type platform      //平台的所有人
24. )**const**;

返回从起始周期到截至周期的平台收益详情，vector<Platform\_Period\_Profit\_Detail>。

示例：*get\_platform\_profits\_detail 1 5 platform\_account1*

## 5. 新增获取作者收益

1. **struct** Poster\_Period\_Profit\_Detail  //作者周期收益详情
2. {
3. uint32\_t                               cur\_period;           //当前周期号
4. account\_uid\_type                       poster\_account;       //作者
6. share\_type                             total\_forward = 0;    //每周期文章转发的收益和
7. flat\_map<asset\_aid\_type, share\_type>   total\_rewards;        //每周期文章打赏的收益和
8. share\_type                             total\_post\_award = 0; //每周期来自内容资金池的收益
10. vector< Post\_Object\_Index >            active\_post\_pids;     //所有有收益的文章id表
11. };
13. vector<Poster\_Period\_Profit\_Detail> get\_poster\_profits\_detail(**const** uint32\_t         begin\_period, //起始周期号
14. **const** uint32\_t         end\_period,   //截至周期号
15. **const** account\_uid\_type poster        //作者
16. )**const**;

返回从起始周期到截至周期的作者收益详情，vector<Poster\_Period\_Profit\_Detail>。

示例：*get\_poster\_profits\_detail 1 5 auther\_account*

# Wallet接口修改整理

## 1. 新增transfer\_extension

1. signed\_transaction transfer\_extension(string from,                //转出人的id或者名字
2. string to,                  //转入人的id或者名字
3. string amount,              //交易数额
4. string asset\_symbol,        //交易的资产
5. string memo,                //交易的备注
6. **bool** isfrom\_balance = **true**, //是否从余额转出。true，从余额转出；false，从零钱转出
7. **bool** isto\_balance = **true**,   //是否转到余额。true，转到余额；false，转到零钱
8. **bool** csaf\_fee = **true**,       //使用积分付手续费。true，积分付手续费；false，余额付手续费
9. **bool** broadcast = **false**      //是否广播
10. );

更新说明：transfer\_extension 用于支持账户之间的余额、零钱的相互交易。

示例：*transfer\_extension test\_account1 test\_account1 5000 YOYO "" true false true true*

## 2. 扩展account\_auth\_platform

1. signed\_transaction account\_auth\_platform(string account,              //账户的id或名字
2. string platform\_owner,       //平台的账户id或名字
3. string limit\_for\_platform,   //授予平台零钱的使用限额
4. uint32\_t permission\_flags =  //授予平台的操作权限
5. account\_statistics\_object::Platform\_Permission\_Forward |  //转帖权
6. account\_statistics\_object::Platform\_Permission\_Liked |    //点赞权
7. account\_statistics\_object::Platform\_Permission\_Buyout |   //买断权
8. account\_statistics\_object::Platform\_Permission\_Comment |  //评论权
9. account\_statistics\_object::Platform\_Permission\_Reward |   //打赏权
10. account\_statistics\_object::Platform\_Permission\_Post,      //发贴权
11. **bool** csaf\_fee  = **true**,       //使用积分付手续费。true，积分付手续费；false，余额付手续费
12. **bool** broadcast = **false**       //是否广播
13. );

更新说明：相对于原来的account\_auth\_platform增加了，账户授予平台的零钱使用限额，以及授予平台的各类操作权限。

示例：*account\_auth\_platform test\_account8 platform\_account1 10000 255 true true*

## 3. 新增create\_post

1. **struct** post\_create\_ext
2. {
3. uint8\_t                                           post\_type = post\_operation::Post\_Type\_Post; //文章类型
4. optional<std::string>                             forward\_price;                              //转发价格
5. optional< map<account\_uid\_type, receiptor\_ext> >  receiptors;                                 //受益人列表
6. optional<license\_lid\_type>                        license\_lid;                                //文章许可id
7. uint32\_t                                          permission\_flags =                          //文章开放的权限
8. post\_object::Post\_Permission\_Forward |  //文章允许转发
9. post\_object::Post\_Permission\_Liked |    //文章允许点赞
10. post\_object::Post\_Permission\_Buyout |   //文章允许买断
11. post\_object::Post\_Permission\_Comment |  //文章允许评论
12. post\_object::Post\_Permission\_Reward;    //文章允许打赏
13. };
15. signed\_transaction create\_post(string           platform,               //文章发布的平台账户的id或名字
16. string           poster,                 //文章作者的账户id或者名字
17. string           hash\_value,             //文章hash
18. string           title,                  //文章标题
19. string           body,                   //文章内容
20. string           extra\_data,             //文章额外内容
21. string           origin\_platform = "",   //文章的原平台
22. string           origin\_poster = "",     //文章的原作者
23. string           origin\_post\_pid = "",   //文章的原id
24. post\_create\_ext  ext = post\_create\_ext(),//扩展结构，参见上文结构体
25. **bool** csaf\_fee  = **true**,                   //使用积分付手续费。true，积分付手续费；false，余额付手续费
26. **bool** broadcast = **false**                   //是否广播
27. );

更新说明：新增create\_post用于创建文章，回复文章，转发文章等等。

示例：*create\_post platform\_account1 test\_account1 987456321 postname postbody postextry "" "" "" {post\_type :0, forward\_price :null, receiptors :null, license\_lid :1, permission\_flags :255} true true*

## 4. 新增update\_post

1. **struct** post\_update\_ext
2. {
3. optional<std::string>       forward\_price;     //更新转发价，需要作者签名
4. optional<std::string>       receiptor;         //更新的目标受益人，收益人相关内容更新，需要该收益人签名
5. optional<**bool**>              to\_buyout;         //更新是否出售收益，需要目标收益人签名
6. optional<uint16\_t>          buyout\_ratio;      //更新出售收益比率，需要目标收益人签名
7. optional<std::string>       buyout\_price;      //更新出售收益价格，需要目标收益人签名
8. optional<time\_point\_sec>    buyout\_expiration; //更新出售收益过期时间，需要目标收益人签名
9. optional<license\_lid\_type>  license\_lid;       //更新文章许可id ，需要作者签名
10. optional<uint32\_t>          permission\_flags;  //更新文章开放权限 ，需要作者签名
11. };
13. signed\_transaction update\_post(string           platform,              //文章所在平台
14. string           poster,                //文章作者
15. string           post\_pid,              //文章的id
16. string           hash\_value = "",       //文章的hash
17. string           title = "",            //文章的标题
18. string           body = "",             //文章的内容
19. string           extra\_data = "",       //文章的额外内容
20. post\_update\_ext ext = post\_update\_ext(),//扩展结构，参见上文结构体
21. **bool** csaf\_fee  = **true**,                  //使用积分付手续费。true，积分付手续费；false，余额付手续费
22. **bool** broadcast = **false**                  //是否广播
23. );

更新说明：新增update\_post，用于更新文章内容，开发权限，许可id，转发价以及受益人出售收益的挂单。

示例：*update\_post platform\_account1 test\_account1 2 "" "" "" "" {forward\_price : 500, receiptor : test1, to\_buyout : true, buyout\_ratio : 3000, buyout\_price : 300, buyout\_expiration :* *1546272000, license\_lid : 1, permission\_flags : null} true true*

## 5. 新增score\_a\_post

1. signed\_transaction score\_a\_post(string           from\_account, //点赞打分发起人的账户id或者名字
2. string           platform,     //文章所在平台
3. string           poster,       //文章作者
4. post\_pid\_type    post\_pid,     //文章的id
5. int8\_t           score,        //打的分数，范围-5~5
6. string           csaf,         //花费的积分数
7. **bool** csaf\_fee  = **true**,         //使用积分付手续费。true，积分付手续费；false，余额付手续费
8. **bool** broadcast = **false**         //是否广播
9. );

更新说明：新增score\_a\_post用于账户对文章帖子点赞打分，打分范围为-5~5，需要花费账户积分。

示例：*score\_a\_post test\_account2 platform\_account1 test\_account1 2 5 10 true true*

## 6. 新增reward\_post

1. signed\_transaction reward\_post(string           from\_account, //打赏人的账户id或者名字
2. string           platform,     //文章所在平台
3. string           poster,       //文章作者的账户id或者名字
4. post\_pid\_type    post\_pid,     //文章的id
5. string           amount,       //打赏的金额
6. string           asset\_symbol, //打赏的资产名
7. **bool** csaf\_fee  = **true**,         //使用积分付手续费。true，积分付手续费；false，余额付手续费
8. **bool** broadcast = **false**         //是否广播
9. );

更新说明：新增reward\_post用于个人账户对文章的打赏，打赏可以是链上任意资产。

示例：*reward\_post test\_account3 platform\_account1 test\_account1 2 1000 YOYO true true*

## 7. 新增reward\_post\_proxy\_by\_platform

1. signed\_transaction reward\_post\_proxy\_by\_platform(string           from\_account, //打赏人的账户id或者名字
2. string           platform,     //文章所在平台
3. string           poster,       //文章作者的账户id或者名字
4. post\_pid\_type    post\_pid,     //文章的id
5. string           amount,       //打赏的金额，动用的是授权给平台的零钱，必须是YOYO
6. **bool** csaf\_fee  = **true**,         //使用积分付手续费。true，积分付手续费；false，余额付手续费
7. **bool** broadcast = **false**         //是否广播
8. );

更新说明：新增reward\_post\_proxy\_by\_platform用于平台代理账户对文章进行打赏。动用的是用户授权给平台的零钱，需要相应的授权并消耗授权的零钱额度，打赏资产必须是YOYO。

示例：*reward\_post\_proxy\_by\_platform test\_account3 platform\_account1 test\_account1 2 1000 true true*

## 8. 新增buyout\_post

1. signed\_transaction buyout\_post(string           from\_account,     //买断发起人的账户id或者名字
2. string           platform,         //文章所在平台
3. string           poster,           //文章作者的账户id或者名字
4. post\_pid\_type    post\_pid,         //文章的id
5. string           receiptor\_account,//买断的受益人的账户id或者名字
6. **bool** csaf\_fee  = **true**,             //使用积分付手续费。true，积分付手续费；false，余额付手续费
7. **bool** broadcast = **false**             //是否广播
8. );

更新说明：新增buyout\_post用于买断文章受益人的收益比。其中买断的受益人receiptor\_account，必须先完成出售文章受益的有效挂单，参见update\_post操作。

示例：*buyout\_post test\_account3 platform\_account1 test\_account1 2 test\_account1 true true*

## 9. 新增create\_license

1. signed\_transaction create\_license(string           platform,     //创建的平台的账户id或者名字
2. uint8\_t          license\_type, //许可的类型
3. string           hash\_value,   //许可的hash
4. string           title,        //许可的标题
5. string           body,         //许可的内容
6. string           extra\_data,   //许可的额外内容
7. **bool** csaf\_fee  = **true**,         //使用积分付手续费。true，积分付手续费；false，余额付手续费
8. **bool** broadcast = **false**         //是否广播
9. );

更新说明：新增create\_license用于平台账户创建许可。仅有平台账户才可以创建许可，且许可不可修改。

示例：*create\_license platform\_account1 1 123456789 license\_one licensebody licenseextra true true*

## 10. 新增account\_manage

1. **struct** opt
2. {
3. optional< **bool** > can\_post;  //允许发帖
4. optional< **bool** > can\_reply; //允许回复
5. optional< **bool** > can\_rate;  //允许点赞
6. };
8. signed\_transaction account\_manage(string executor,                       //管理员账户的id或名字
9. string account,                        //管理的账户
10. account\_manage\_operation::opt options, //扩展结构，参见上文结构体
11. **bool** csaf\_fee  = **true**,                 //使用积分付手续费。true，积分付手续费；false，余额付手续费
12. **bool** broadcast = **false**                 //是否广播
13. );

更新说明：新增account\_manage用于账户的管理。

示例：*account\_manage init test\_account1 {can\_post : null, can\_reply : true, can\_rate : true} true true*

## 11. 新增get\_post

1. post\_object get\_post(string platform\_owner, //文章发布的平台账户的id或名字
2. string poster\_uid,     //文章作者的账户id或者名字
3. string post\_pid        //文章的id
4. );

更新说明：新增get\_post用于post\_object的查询。

示例：*get\_post platform\_account1 test\_account1 1*

## 12. 新增get\_score

1. score\_object get\_score(string platform,     //文章所在平台
2. string poster\_uid,   //文章作者的账户id或者名字
3. string post\_pid,     //文章的id
4. string from\_account  //点赞打分的账户id或者名字
5. );

更新说明：新增get\_score用于score\_object的查询。

示例：*get\_score platform\_account1 test\_account1 1 test\_account2*

## 13. 新增get\_license

1. license\_object get\_license(string platform,   //创建许可的平台的账户id或者名字
2. string license\_lid //许可的id
3. );

更新说明：新增get\_license用于license\_object的查询。

示例：*get\_license platform\_account1 1*

## 14. 新增get\_global\_properties\_extensions

1. get\_global\_properties\_extensions();

更新说明：新增get\_global\_properties\_extensions用于扩展理事会参数查询

示例：*get\_global\_properties\_extensions*

## 15. 更新committee\_proposal\_create扩展参数示例

理事会扩展参数提案创建示例：

*committee\_proposal\_create init1 [[3 , {content\_award\_interval : 300,platform\_award\_interval : 300,max\_csaf\_per\_approval : 1000,approval\_expiration : 31536000,min\_effective\_csaf : 10,total\_content\_award\_amount : 10000000000000,total\_platform\_content\_award\_amount : 10000000000000,total\_platform\_voted\_award\_amount : 10000000000000,platform\_award\_min\_votes : 1,platform\_award\_requested\_rank : 100, platform\_award\_basic\_rate : 3000, casf\_modulus : 30000, post\_award\_expiration : 62000000, approval\_casf\_min\_weight : 4000, approval\_casf\_first\_rate : 2000, approval\_casf\_second\_rate : 8000, receiptor\_award\_modulus : 9000, disapprove\_award\_modulus : 11000}]] 200 1 200 200 true true*

## 16. 新增get\_posts\_by\_platform\_poster

1. vector<post\_object> get\_posts\_by\_platform\_poster(string                                    platform\_owner,   //文章发布的平台账户的id或名字
2. string                                    poster,           //文章作者的账户id或者名字
3. time\_point\_sec                            begin\_time\_range, //开始时间
4. time\_point\_sec                            end\_time\_range,   //结束时间
5. uint32\_t                                  limit             //展示数目限制
6. );

更新说明：新增get\_posts\_by\_platform\_poster，用于通过作者查询所有作者时间限制内发布的文章。

示例：*get\_posts\_by\_platform\_poster platform\_account1 test\_account1 1546272000 1547026795 100*

## 17. 新增list\_scores

1. vector<score\_object> list\_scores(string   platform,               //文章所在平台的账户id或者名字
2. string   poster\_uid,             //文章作者的账户id或者名字
3. string   post\_pid,               //文章的id
4. uint32\_t limit,                  //展示数目限制
5. **bool**     list\_cur\_period = **true**  //是否仅展示当前收益分配周期的点赞。true：只展示当前周期点赞；false：展示所有点赞。
6. );

更新说明：新增list\_scores，用于通过文章来查询所有的当前收益分配周期或者所有的点赞信息。

示例：*list\_scores platform\_account1 test\_account1 1 1000 true*

## 18. 新增list\_licenses

1. vector<license\_object> list\_licenses(string platform, //平台的账户id或名字
2. uint32\_t limit   //展示数目限制
3. );

更新说明：新增list\_licenses，用于通过平台查询所有该平台发布的许可信息。

示例：*list\_licenses platform\_account1 100*

## 19. 新增get\_account\_statistics

1. account\_statistics\_object get\_account\_statistics(string account // 账户的id或者名称
2. );

更新说明：新增get\_account\_statistics，用于查看账户的统计信息。

示例：*get\_account\_statistics test\_account1*

## 20. 新增get\_post\_profits\_detail

1. vector<active\_post\_object> get\_post\_profits\_detail(uint32\_t         begin\_period, //起始周期号
2. uint32\_t         end\_period,   //截至周期号
3. string           platform,     //文章所在平台的账户id或名字
4. string           poster,       //文章的作者的账户id或名字
5. string           post\_pid      //文章的id
6. );

更新说明：查询从起始周期到截至周期的文章的所有收益详情。

示例：*get\_post\_profits\_detail 1 5 platform\_account1 auther\_account 1*

## 21. 新增get\_platform\_profits\_detail

1. vector<Platform\_Period\_Profit\_Detail> get\_platform\_profits\_detail(uint32\_t         begin\_period, //起始周期号
2. uint32\_t         end\_period,   //截至周期号
3. string           platform      //平台的账户id或名字
4. );

更新说明：查询从起始周期到截至周期的平台的所有收益详情。

示例：*get\_platform\_profits\_detail 1 5 platform\_account1*

## 22. 新增get\_poster\_profits\_detail

1. vector<Poster\_Period\_Profit\_Detail> get\_poster\_profits\_detail(uint32\_t         begin\_period, //起始周期号
2. uint32\_t         end\_period,   //截至周期号
3. string           poster        //作者的账户id或名字
4. );

更新说明：查询从起始周期到截至周期的作者的所有收益详情。

示例：*get\_poster\_profits\_detail 1 5 auther\_account*

## 23. 新增get\_score\_profit

1. share\_type get\_score\_profit(string    account, //账户的名称或者id
2. uint32\_t  period   //收益分配的周期号
3. );

更新说明：查询某账户在某收益周期的点赞总收入

示例：*get\_score\_profit test\_account1 1*

# 命令行选项

## 新增active-post-periods

更新说明： 新增active-post-periods命令行参数，用于设置数据库保存最近多少内容评奖周期的活跃文章对象，默认保存最近十期。

示例：*--active-post-periods=20*

# 评奖规则

## 平台资金池（内容）

可参与平台资金池（内容）评奖条件：

平台在评奖周期内有评分行为(即该平台内文章的打分消耗的积分总数大于0)

平台资金池（内容）分配：

根据可参与评奖平台的所有积分数（评奖周期内）占所有平台的积分数总和（评奖周期内）比例分配

## 2.平台资金池（投票）

可参与平台资金池（投票）评奖条件：

1. 平台投票数不小于平台参与评奖所需最小投票数（理事会参数）；
2. 平台投票数排名在平台参与评奖所需排名数（理事会参数）之内；

平台资金池（投票）分配：

平台资金池的分配基础比例（理事会参数）平均分配给可参与评奖平台；  
剩余根据可参与评奖平台的投票数占所有可参与评奖平台的投票数总和比例分配；

## 3.内容资金池

参考以下流程图（单独文件\yoyow-core\tests\文章奖池流程图）

有效文章1

有效文章2

有效文章N

计算M = M1+M2+M3+…+Mn；根据每篇有效文章的Mi占M比例分配文章奖池

25%

75%

该文章第一次参与分配？

不分配

否

计算评分者i有效积分Si;

计算所有评分者有效积分和：S = S1+S2+S3+…+Sn

是

根据评分者i有效积分Si与总有效积分S占比计算收益Yi

活跃文章i

该文章总积分 >= 文章参与评奖最小积分（理事会参数）

不参与评奖

Mi >= 0 ?

计算公式1：Mi = 该文章积分总数 + ∑（评分\*消耗积分\*系数（理事会参数）/5）

不参与评奖

否

否

是

是

文章收入：按文章收益列表占比分配

（先分配账户，剩余分配给平台）

计算公式2：

前30%（理事会参数）的积分：100%权重；

30%~70%（理事会参数）：权重逐渐由100%降至50%（理事会参数）；后30%：固定50%

评分者i增加收益Yi

评分者i收益增加Yi\*反正者收益比例（理事会参数）

该篇文章是否赞大于踩即∑（评分\*消耗积分\*系数（理事会参数）/5 ）>= 0 ?

是

评分者i是否点赞即评分 >= 0 ?

是

否

否

该篇文章是否赞大于踩即∑（评分\*消耗积分\*系数（理事会参数）/5 ）>= 0 ?

是

否

文章收入\*文章受益人奖励系数（理事会参数）：按文章收益列表占比分配

（先分配账户，剩余分配给平台）

以评分者的第一个积分所在的权重点为该评分者消耗的所有积分权重

举例：评分者A、B、C先后都花费1000积分点赞；

A的所有积分权重为100%，有效积分为1000

B的所有积分权重为95%，有效积分为950

C的所有积分权重为54%，有效积分为540

总共有效积分为1000+950+540