

# 고급 등급 페이지


## 개요

이 페이지를 사용하여 트래커 항목 혹은 위키 페이지를 평가하기 위한 "등급" 시스템을 구성하십시오.

## 관련 토픽

- 전자 민주주의 시스템
- 버그 및 희망사항
- 등급

## 접근하려면

관리 패널 상의 등급 아이콘  을 클릭하십시오  
또는  
<http://귀하도메인주소.com/tiki-admin.php?page=rating> 으로 접근하십시오

## 주의

티키는 현재 고급 등급을 통하여 분류하는 것을 다음의 기능들에서 지원합니다:

- 기사
- 위키
- 댓글

또한 수학적 연산 트래커 필드 를 참조하십시오

### Advanced Rating

Global configuration

☒ Advanced Rating

Rating recalculation mode: Recalculate on vote

Wiki

☒ Simple wiki ratings

Wiki rating options: 1,2,3,4,5

Articles

☒ User ratings on articles

Article rating options: 1,2,3,4,5

Apply

Create New

Name

Create

고급 등급 페이지

설정	설명	Default
전역 구성		
고급 등급	내부 등급 시스템을 활성화함, 이는 트래커, 기사 혹은 기타 기능에서의 값들을 계산하는데 사용됨.	

설정	설명	Default
등급 재계산 모드:	<p>등급 합계가 언제 그리고 어떻게 재계산되는지를 결정:</p> <p>* <b>투표 시</b> (기본값): 투표가 수행될 때 매번마다 개체에 대한 점수가 재계산되어야함을 지정. 이 선택사항은 규모가 더 적은 사이트 및 등급이 사용될 때 비교적 단순한 연산 방식에 적절함.</p> <p>* <b>로드 시 무작위</b>: 이는 무작위 기반으로 페이지 로드 시에 일부 소수의 점수들이 계산되도록 will cause a few scores to be calculates on page load on a random basis (odds and count can be configured to adapt to site load). This option is suitable for calculation rules involving time that must be recalculated even if no new votes occurred.</p> <p>* <b>Random on vote</b> is similar to random on load, but will recalculate multiple scores (not necessarily including the current object) when a vote is performed. It is suitable for similar situations. The best option will depend on site load.</p> <p>* <b>Periodic</b>: is the best option for heavy load sites, making sure all calculations are done outside the web requests. A cron job must be set-up manually by the site's administrator. A sample script is available at the end of this page.</p> <p>Depending on the site load, some options may be better than others; on large volume sites, we recommend <b>cron job</b>. The <b>Recalculate on vote</b> recalculation may be inaccurate if rating calculation depends time.</p> <p><b>Before any attempt to re-index the object:</b> Ties into the <a href="#">Search and List from Unified Index</a> and updates the calculation at index-time.</p>	Recalculate on vote
Recalculation odds (1 in X):		
Recalculation count:		
<b>Wiki</b>		
Simple wiki ratings	Enable a simple rating bar at the top of each wiki page.	
Wiki rating options:	List of options for the simple wiki ratings.	1,2,3,4,5
<b>Articles</b>	Enable a simple rating bar at the top of each articles page.	
User ratings on articles		
Article rating options:		

The feature must first be enabled through this same administration panel. Along with the feature, a few options are available. Among them, the score recalculation period must be defined. These are the available options:

- **On vote (default)** indicates that the score for the object should be recalculated every time a vote is performed. This option is suitable for sites with lower volumes and relatively simple calculation methods when ratings are used.
- **Random on load** will cause a few scores to be calculates on page load on a random basis (odds and count can be configured to adapt to site load). This option is suitable for calculation rules involving time that must be recalculated even if no new votes occurred.
- **Random on vote** is similar to random on load, but will recalculate multiple scores (not necessarily including the current object) when a vote is performed. It is suitable for similar situations. The best option will depend on site load.
- **Periodic** is the best option for heavy load sites, making sure all calculations are done outside the web requests. A cron job must be set-up manually by the site's administrator. A sample script is available at the end of this page.

For the random options, the odds of recalculating must be specified as a dice roll. For each occurrence of a recalculation, a limit to how many scores can be calculated must be specified to avoid the hang-up effect on the page load.

The value ranges for each object type can also be specified through the administration panels.

The common *sort\_mode* parameter to lists can be used to activate sorting using advanced ratings. To do so, the sort mode must be set to *adv\_rating\_X\_asc* or *adv\_rating\_X\_desc* where *X* is the ID of the rating configuration. The default sort can also be set to advanced ratings in the administration panel where applicable.

## Calculation configuration

From the administration panel, new calculations can be added. Initially, only the name is required. When created, the calculation will contain suitable default values.

For wiki pages:



Thus, visitors can provide feedback like:

- Did this page help you solve the issue?
- Was this page easy to understand?

---

## Advanced Rating

Introduced in [Tiki5](#), the advanced rating feature allows for more control over the aggregation of scores.

Rating methods are defined globally and will be used for all supported objects. They are defined through the **Advanced Rating** administration panel ([tiki-admin.php?page=rating](#)). Multiple methods can be created. If a method contains type-specific calculations, it will be ignored when performing the calculation.

Features currently supporting sorting through advanced rating:

- [Articles](#)
- [Wiki](#)
- [Comments](#)

## Sorting items according to advanced rating

Note that the sort mode to use when needing to sort by advanced rating is either *adv\_rating\_xx\_asc* or *adv\_rating\_xx\_desc*, where *xx* is the `ratingConfigId`.

Feature request: Can we make this take the name of the config instead of the `ratingConfigId` as well?

## Set-up

By default, each calculated value is kept for 1 hour (3600 seconds). This limit does not apply when recalculating on vote, but is used for every other technique to avoid recalculating the same scores over and over again.

The calculation is defined as a small piece of code, similar to functional languages, which is very close to mathematical representations. Creating custom formulas is expected to require some mathematical skills. However, this documentation should provide examples for most frequent cases.

The editor in the administration panel performs extensive validation and will make it impossible to save the formula unless it can be evaluated. Checks are performed for:

- Syntax errors
- Unknown functions
- Missing arguments
- Invalid argument values
- Unknown input variables

#### Default formula

```
(rating-average (object type object-id))
```

It can be altered to limit the vote consideration to a limited time span, 30 days for example.

#### Recent votes only

```
(rating-average (object type object-id) (range (mul 3600 24 30)) )
```

In the language, spaces do not matter. Only the parenthesis indicate structure. **rating-average** is a function that fetches the ratings for a given object. *type* and *object-id* are standard variables fed when calculating a rating. **object** and **range** are configuration options of the function.

**mul** is a mathematical function. (mul 3600 24 30) is equivalent to 3600\*24\*30.

The functions can be combined in various ways. For example, we could calculate a score that considers the votes from the past month, but gives extra emphasis on the recent ones.

#### Combined vote duration

```
(add (rating-average (object type object-id) (range (mul 3600 24 30))) (rating-average (object type object-id) (range (mul 3600 24 7)))) )
```

Even though the votes are 1-5, the final score can be on an entirely different scale. The language is also extensible if the calculation needs to be combined with other factors or weight. See [Rating Language](#).

All available options are documented in the following section.

## General Reference

### comment

Any comment block is stripped from the formula at parse-time

#### Examples

```
(mul 1 2 (comment Simple enough?)) -> 2
```

### mul (Multiply)

Performs a simple multiplication accepting multiple input values.

#### Examples

```
(mul 3 4) -> 12 (mul (mul 3 4) 5) -> 60 (mul 3 4 5) -> 60 (mul 4 0.5) -> 2
```

## div (Divide)

Performs a simple division accepting multiple input values.

### Examples

```
(div 3 4) -> 0.75 (div (mul 3 10) 5) -> 6 (div 30 5 3) -> 2 (div 4 0.5) -> 8
```

## add (Sum)

Performs a simple sum accepting multiple input

### Examples

```
(add 3 4) -> 7 (add (add 3 4) 5) -> 12 (add 3 4 5) -> 12 (add 4 0.5) -> 4.5
```

## sub (Subtract)

Performs a simple subtraction accepting multiple input

### Examples

```
(sub 3 4) -> -1 (sub (sub 3 4) 5) -> -6 (sub 3 4 5) -> -12 (add 4 0.5) -> 3.5
```

## round

Rounds to a specific number of digits (new in Tiki12)

### Examples

```
(round 4.556234342234 2) -> 4.56 (round 4.556234342234) -> 5
```

## coalesce

Returns the first non-empty value from the list.

### Examples

```
(coalesce 3 4) -> -3 (coalesce (sub 3 3) 5) -> 5 (coalesce 0 0 (str) -10) -> -10 (coalesce 0 0 0 0 0) -> 0
```

## str

Generates a static string when needed and the processor attempts to process the string as a variable. Any arguments will be concatenated using spaces.

**Note:** The quoted string syntax was included in [Tiki13](#).

### Examples

```
(str hello-world) -> "hello-world" (str hello world) -> "hello world" (str hello world foobar) -> "hello world foobar" (str (mul 2 3) "= 6") -> "6 = 6"
```

## concat

Concatenates a string of text. (new in Tiki12)

**Note:** The quoted string syntax was included in [Tiki13](#).

### Examples

```
(concat (str $) 1234 ) -> "$1234" (concat 14 (str %) ) -> "14%" (concat 14 "%") -> "14%"
```

## map

Generates a map (or dictionary).

### Examples

```
(map (key1 1) (key2 2) (key3 (str value3)) ) -> {"key1": 1, "key2": 2, "key3": "value3"}
```

## equals

Compares multiple values.

### Examples

```
(equals 2 (add 1 1) (sub 4 2)) -> 1 (equivalent of 2 == 1+1 && 2 == 4-2) (equals (add 1 1) 3) -> 0
```

## if

Conditionally evaluates a branch.

### Examples

```
(if (equals 2 2) 42 -1) -> 42 (if (equals 2 1) 42 -1) -> -1
```

## and

Ensures all elements evaluate to true.

### Examples

```
(and 3 2 1 2 3) -> 1 (and 2 3 0 2) -> 0
```

## or

Ensures that at least one element evaluates to true. Elements are evaluated sequentially until a false element is found. Others are left unevaluated.

### Examples

```
(or 3 2 1 2 3) -> 1 (or 2 3 0 2) -> 1 (or 0 0) -> 0
```

## hash

Generates a hash based on multiple values. Used primarily to generate aggregate hashes in the [PluginActivityStream](#). Note that because it is a hash, the exact value coming out does not matter. Only that given the same parameter, it will produce the same value.

### Examples

```
(hash 1) -> [sha1("1")] (hash 1 2 3 4) -> [sha1("1/2/3/4")] (hash 1 2 (map (a 3) (b 4))) -> [sha1("1/2/3/4")]
```

## avg

Calculates the average of multiple values. All entries in the list will be flattened if arrays are present.

### Examples

```
(avg 1 2 3) -> 2 ... given list contains [1, 2, 3] (avg list) -> 2
```

## split-list

Produces a multi-dimensional array out of a text string. Each line is expected to be an independent value, each line will be split by a separator into the specified keys.

### Examples

```
... given str contains a list of 3 comma-separated values (split-list (content str) (separator ,) (keys a b c)) -> [{a: 1, b: 2, c: 3}, {a: 2, b: 3, c: 4}]
```

## for-each

For a list of value pairs, such as the output of *split-list*, evaluates a formula for each set of values, returns the list of results.

Within the formula, variables coming from the list will be used first. Fallback will be on the other variables available in the execution context.

### Examples

```
... given items contains [{a: 1, b: 2, c: 3}, {a: 2, b: 3, c: 4}] (for-each (list items) (formula (mul a b c))) -> [6, 24] ... given items contains [{a: 1, b: 2, c: 3}, {a: 2, b: 3, c: 4}] ... and d contains 10 (for-each (list items) (formula (mul c d))) -> [30, 40]
```

## 고급 등급-특정 참조

### rating-average (등급-평균) 및 rating-sum (등급 합계)

등급 함수는 등급 기록 테이블에서 점수를 계산합니다. 사이트에 수행된 각 등급은 데이터베이스 내에 보관되며 사용자 지정 등급을 계산하기 위해서 사용될 수 있습니다. 다양한 선택사항이 문서의 품질 향상을 지원하거나 피드 수집기로 들어오는 데이터의 순위를 매기기 위한 등의 사이트 상의 중요성을 반영하기 위하여 점수를 계산하도록 적용됩니다.

- **object (개체)**, 필수이며 항상 이 내용상 (*개체 유형 개체ID*).
- **range (범위)**, to limit how long votes are considered. Argument is provided as a number of seconds.
- **ignore (무시)**, with *anonymous* as an argument to only consider votes from registered users.
- **keep (유지)**, to only consider one vote per visitor. Unless the option is present, all of the votes are taken into account. The option can be either *latest* or *oldest* to indicate which one to keep.

- **revote (재투표)** can be specified if **keep** is specified. Indicates the time period required between votes. For example, users could be allowed to vote more than once per day, but only their latest vote each day would be considered, if revote is set to mul(24 3600). If the user voted yesterday as well as today, both votes will be counted.

## article-info

계산에 포함될 수 있도록 기사에서 정보를 추출. 첫 번째 인자는 언제나 항상 'article' (기사)가 되어야함. 그 외 다른 값이 되는 경우, 연산은 평가된 개체에 대하여 건너뛰어지게 되어, 이를 공식 유형-특화시켜 버립니다.

사용가능한 속성들:

- rating (등급), 기사에 첨부된 정적 등급
- view-count
- age-second
- age-hour
- age-day
- age-week
- age-month

예제

(article-info type object-id rating) (article-info (str article) 42 age-month)

## 속성

보통의 개체 속성에서 정보를 추출함.

예제

(attribute (object type object-id) (property tiki.proposal.accept) ) -> [value for page in a rating calculation] (attribute (object (str wiki page) 14) (property tiki.proposal.accept) (default 0) ) -> [value for page id 14]

## tracker-field

트래커 항목에서 정보를 추출함. 필드 값은 숫자로 자동으로 변환됨. 값이 발견되지 않거나 적용 불가능 한 경우 0 이 제공됨.

예제

(tracker-field (object type object-id) (field priority) ) -> [value contained in the tracker item field with permanent name "priority"]

## category-present

개체상에 존재하는 모든 나열된 범주의 점수에 1 을 부여함.

예제

(category-present (object type object-id) (list 3 4) ) -> [0, 1 or 2 - Depending on how many of categories 3 or 4 are on the object]



# 부록

통합 검색 (unified search) 이 사용될 때, 재계산은 재색인 동안 이루어지도록 구성될 수 있습니다, 고로 이 스크립트가 필요 없어집니다.

## Cron job

```
<?php chdir('/path/to/tikiroot'); require_once 'tiki-setup.php'; require_once 'lib/rating/ratinglib.php'; $ratinglib->refresh_all();
```

# 단순 위키 등급

# 관련

- [등급](#)
- [Rating Revamp](#)

## 별칭

---

[고급+등급](#) | [고급등급](#)