Multi-Language Support for Khan Academy Platform

(Draft: 30 August, 2012)

Introduction:

The purpose of this paper is to present our understanding of the architecture of the Khan Academy Platform software and to describe how to use the functions provided in this web site to manage the translation of the platform into other languages.

This translation initiative is based on the following principles:

1. get as much of the platform translated as quickly as possible with the least amount of effort (in other words, worry more about getting most translated and leave the most difficult parts to deal with afterwards)
2. enable non-technical users to translate most of the content
3. provide a mechanism to track the status of translation efforts for the various components of the site
4. provide a mechanism for tracking changes to the main Khan Academy site to enable the translations to stay current

This paper is structured as follows:

1. Basic Architecture of the Exercise Portion of the Khan Academy Platform
    a. Relationship of the Knowledge Map to the Exercise Files
    b. Basic Structure of the Khan Academy Exercise Platform Software
    c. Structure of the Typical Khan Academy Exercise File
    d. How Function Files Generate Text in Exercises
2. Website to Support the Translation of the Khan Academy Exercise Platform
    a. What Needs to be Translated
    b. Overview of The Translation Process
    c. Step 1: Create Template Records (authorized users only)
    d. Step 2: Tokenize Files and Create Token Records (authorized users only)
    e. Step 3: Cloning Tokens to Another Language
    f. Step 4: Generated Translated Template for the Applicable Language
    g. Step 5: View Translated Template for the Applicable Language
    h. Try Out the Exercise
    i. Step 6: Export Translated Site
3. Conclusion

In addition to the functions related to translating the exercise platform, the site also includes a place for advocates of better education to exchange ideas and collaborate on issues of mutual interest affecting improving educational achievement, including:

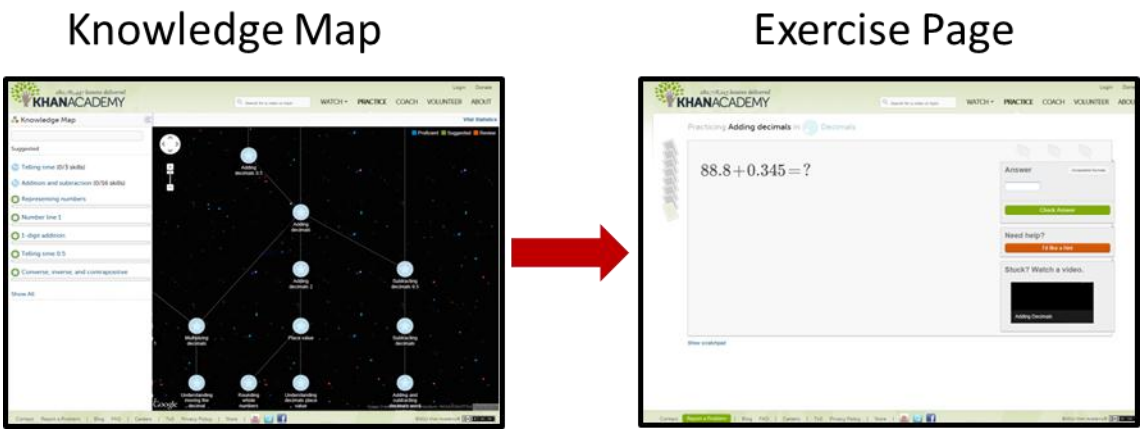
1. cataloging or arranging availability of learning resources such as videos, games, books, etc., in their language that are compatible with the needs of their geographic area (e.g., school districts may teach different methods)
2. Having a Khan Academy or similar learning platform available in their language.

3. Maximizing the number of students benefiting from the platform and content by:
    a. motivating youth to participate
    b. establishing effective alliances and partnerships with those with similar goals or able to help in reaching, motivating or supporting youth learning efforts
    c. Etc.

Basic Architecture of the Exercise Part of the Khan Academy Platform:

<u>Relationship of the Knowledge Map to the Exercise Files</u>

You are familiar with the Practice Menu which displays the Knowledge Map and when you click on a Knowledge Topic it takes you to the Exercise Screen.

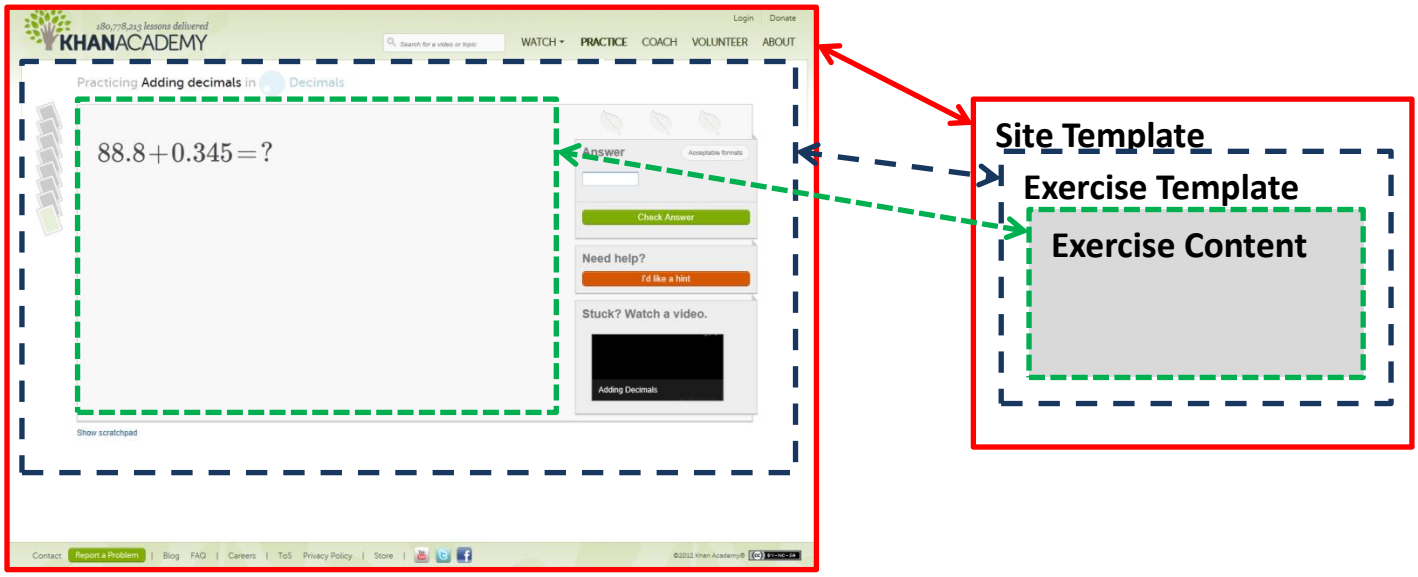


Behind each exercise block, there is an html file that generates the content pertaining to that exercise sequence.

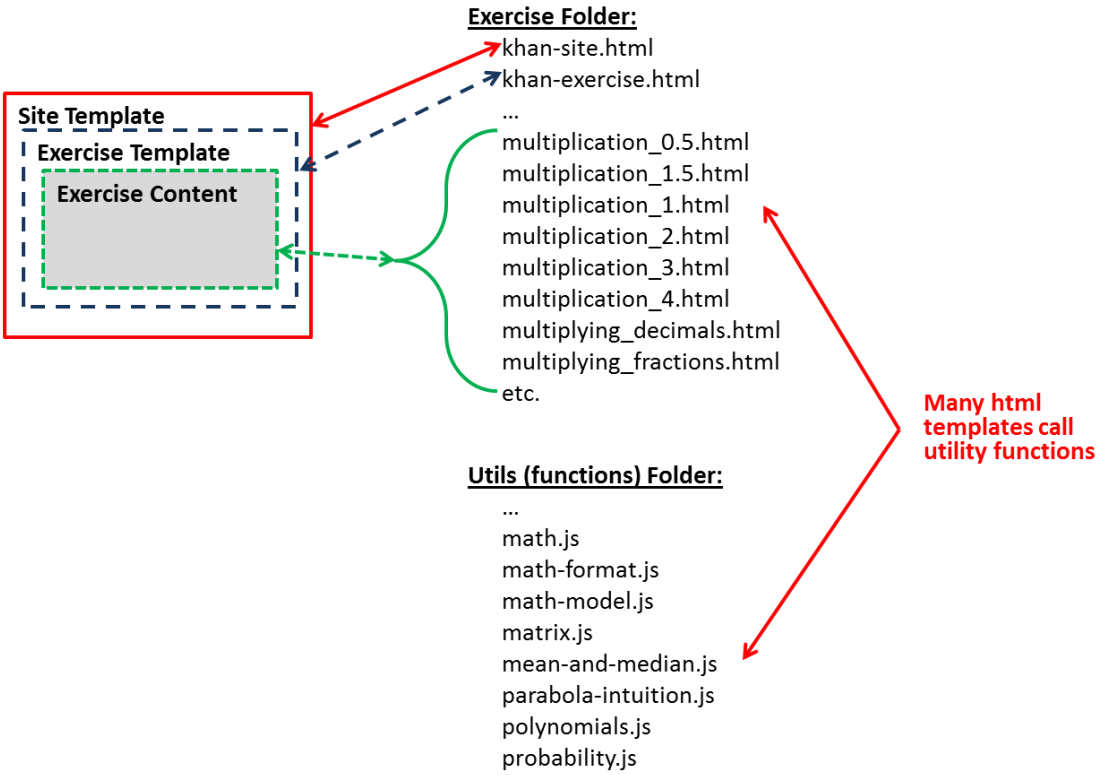
Each of these files generates what I refer to as Exercise Content. (The files consist of simple HTML and Javascript – including calls to functions, etc. that result in the display html that is delivered to the user's browser. For our current purposes, however, it is only important to understand that there are a great many exercise files.)

Basic Structure of the Khan Academy Exercise Platform Software

The Exercise Interface is made up of 3 key structure components:



This translates into a file structure as follows:



Structure of the Typical Khan Academy Exercise File:

At the exercise level, most exercise files have the following sections:

Exercise: rounding_numbers.html

There are 4 sections to most exercises:

Variables:

Question:

Solution:

Hints:

```
<div class="problems">
    <div data-weight="2">
        <div class="vars">
            <var id="DIGITS">shuffle( [1, 2, 3, 4, 5, 6, 7, 8, 9], 5 )</var>
            <var id="NUM">+DIGITS.join( "" )</var>
            <var id="PLACE">randFromArray([ -2, -3 ])</var>
            <var id="TPLACE">placesLeftOfDecimal[ -PLACE ]</var>
            <var id="ROUNDED">roundTo( PLACE, NUM )</var>
        </div>
        <div class="question">
            <p>Round <code><var>commafy( NUM )</var></code> to the nearest <var>TPLACE</var>.</p>
        </div>
        <div class="solution" data-forms="integer, decimal"><var>ROUNDED</var></div>
        <div class="hints">
            <p>There are two ways to think about this problem.</p>
            <p>1st way: Look at the <var>placesLeftOfDecimal[ -1 - PLACE ]</var>s digit <code><var>DIGITS
            <p data-if="DIGITS[ 5 + PLACE ] >= 5">Because it is <span data-if="DIGITS[ 5 + PLACE ] > 5">m
            <code><var>ROUNDED</var></code>.</p>
            <p data-else>Because it is less than <code>5</code>, we round down, giving <code><var>commafy
```

The Question and Hints sections are those most likely to contain content to be translated:

Exercise: rounding_numbers.html

There are a number of text snippets that need to be translated, such as:

Question:

Hints:

```
<div class="problems">
    <div data-weight="2">
        <div class="vars">
            <var id="DIGITS">shuffle( [1, 2, 3, 4, 5, 6, 7, 8, 9], 5 )</var>
            <var id="NUM">+DIGITS.join( "" )</var>
            <var id="PLACE">randFromArray([ -2, -3 ])</var>
            <var id="TPLACE">placesLeftOfDecimal[ -PLACE ]</var>
            <var id="ROUNDED">roundTo( PLACE, NUM )</var>
        </div>
        <div class="question">
            <p>Round <code><var>commafy( NUM )</var></code> to the nearest <var>TPLACE</var>.</p>
        </div>
        <div class="solution" data-forms="integer, decimal"><var>ROUNDED</var></div>
        <div class="hints">
            <p>There are two ways to think about this problem.</p>
            <p>1st way: Look at the <var>placesLeftOfDecimal[ -1 - PLACE ]</var>s digit <code><var>DIGITS
            <p data-if="DIGITS[ 5 + PLACE ] >= 5">Because it is <span data-if="DIGITS[ 5 + PLACE ] > 5">m
            <code><var>ROUNDED</var></code>.</p>
            <p data-else>Because it is less than <code>5</code>, we round down, giving <code><var>commafy
```

How Function Files Generate Text in Exercises:

There may also be text content generated by functions called in the exercise templates:

Exercise: rounding_numbers.html

There are also some calls to functions that return content:

Variables:

Question:

```
<div class="problems">
    <div data-weight="2">
        <div class="vars">
            <var id="DIGITS">shuffle( [1, 2, 3, 4, 5, 6, 7, 8, 9], 5 )</var>
            <var id="NUM">+DIGITS.join( "" )</var>
            <var id="PLACE">randFromArray([ -2, -3 ])</var>
            <var id="TPLACE">placesLeftOfDecimal[ -PLACE ]</var>
            <var id="ROUNDED">roundTo( PLACE, NUM )</var>
        </div>

        <div class="question">
            <p>Round <code><var>commafy( NUM )</var></code> to the nearest <var>TPLACE</var>.</p>
        </div>
```

Math Utility Functions (utils/math.js)

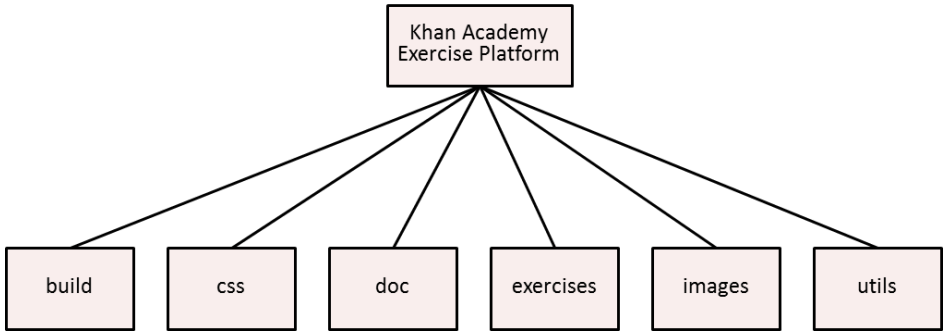
```
placesLeftOfDecimal: ["one", "ten", "hundred", "thousand"],
placesRightOfDecimal: ["one", "tenth", "hundredth", "thousandth"],

powerToPlace: function(power) {
    if (power < 0) {
        return KhanUtil.placesRightOfDecimal[-1 * power];
    } else {
        return KhanUtil.placesLeftOfDecimal[power];
    }
},
```

Text in Function to be translated

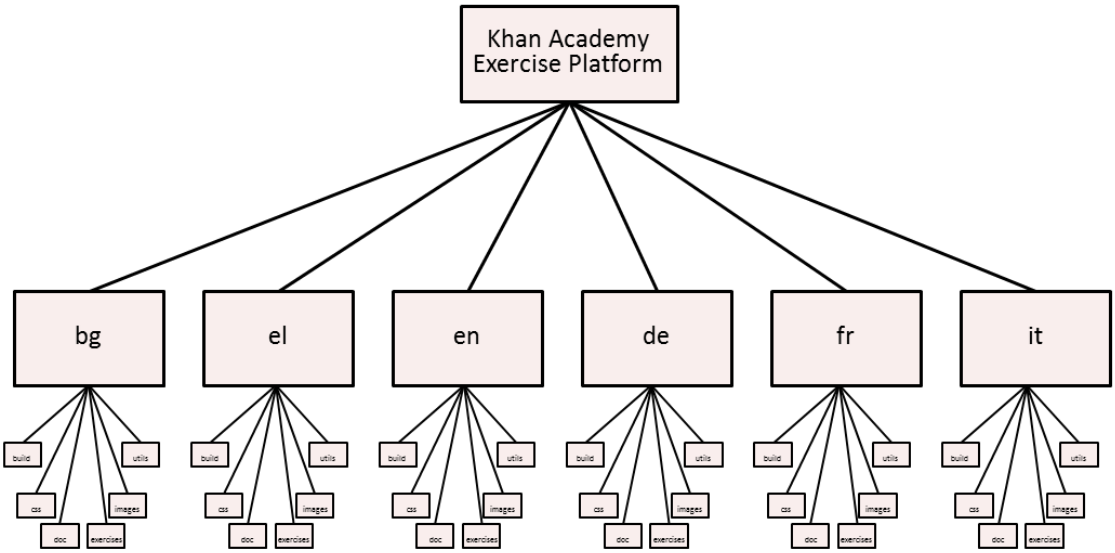
Khan Academy Folder Structure:

Khan Academy Folder Structure



The basic concept for the translation is simply to have a parallel folder structure for each language (although the content in the other languages would be stored in a database and the files in the parallel folder structures would be automatically generated and managed.

Translation is accomplished by having a parallel structure of files for each language



Website to Support the Translation of the Khan Academy Exercise Platform

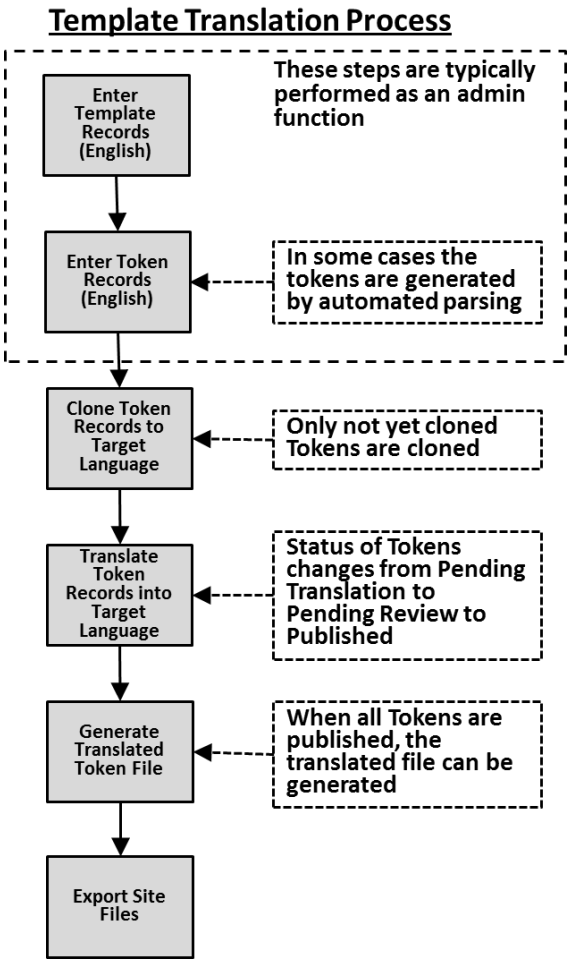
What Needs to be Translated:

Translating an exercise therefore generally consists of translating the following:

1. The visible text in the Site Template (file: khan-site.html) – done once for all exercises.
2. The visible text in the Exercise Template (file: khan-exercise.html) – done once for all exercises.
3. The visible text in the individual exercise files: done once for each applicable exercise (some exercises have no text, and some, e.g. word problem exercises have text that is too complicated for simple translation)
4. The text in some of the utility function (.js) files (in some cases, the exercise files call functions and in some cases those functions return text to the calling exercise.html file).
5. The visible text in the Report Templates (the container templates for the various coach and learner reports)
6. Etc.

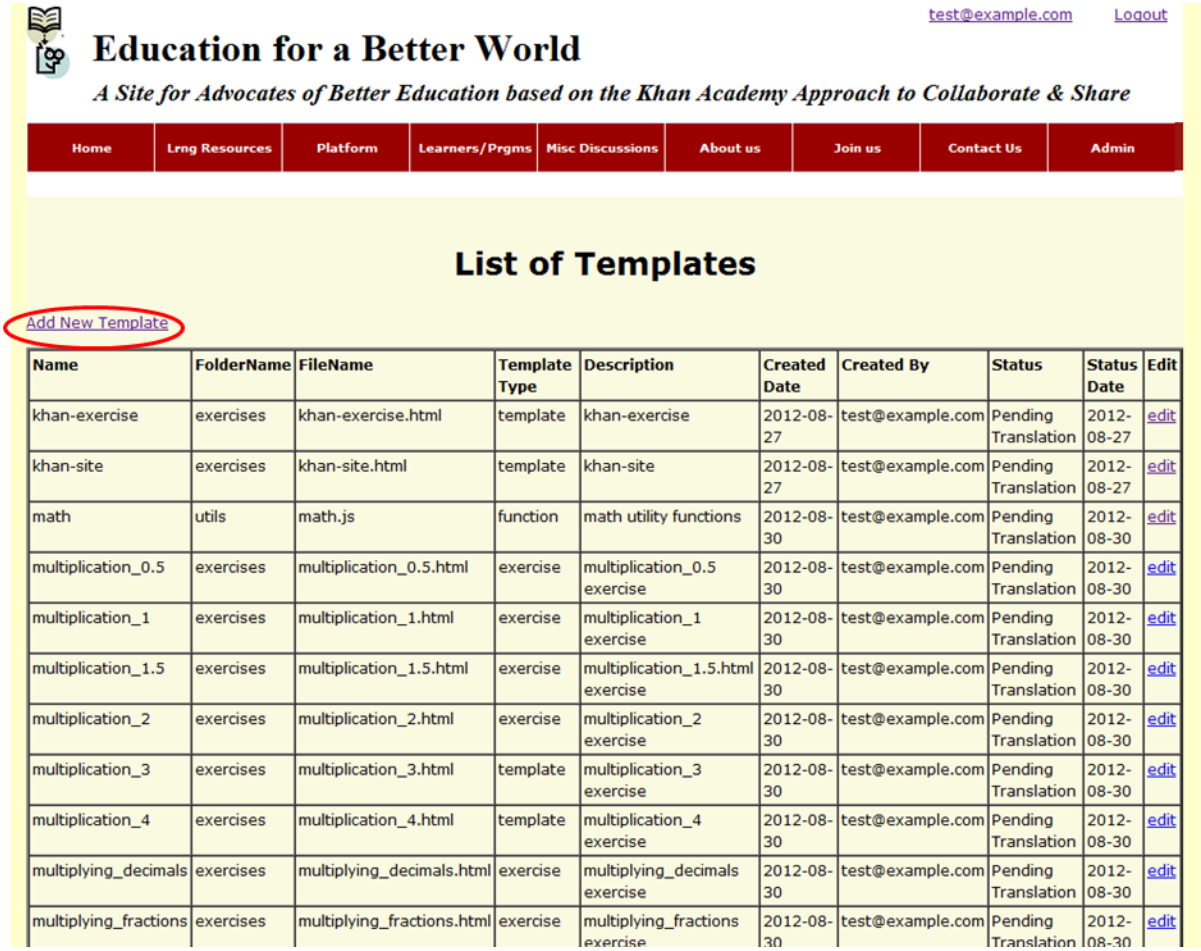
Overview of The Translation Process:

For the purpose of the translation process, templates are created for every file that contains or generates visible text, regardless of type. Conceptually, the translation process is as follows:



Step 1: Create Template Records:

This is an admin function that is available only to a restricted number of users on the site.



test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

List of Templates

Add New Template

Name	FolderName	FileName	Template Type	Description	Created Date	Created By	Status	Status Date	Edit
khan-exercise	exercises	khan-exercise.html	template	khan-exercise	2012-08-27	test@example.com	Pending Translation	2012-08-27	edit
khan-site	exercises	khan-site.html	template	khan-site	2012-08-27	test@example.com	Pending Translation	2012-08-27	edit
math	utils	math.js	function	math utility functions	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplication_0.5	exercises	multiplication_0.5.html	exercise	multiplication_0.5 exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplication_1	exercises	multiplication_1.html	exercise	multiplication_1 exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplication_1.5	exercises	multiplication_1.5.html	exercise	multiplication_1.5.html exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplication_2	exercises	multiplication_2.html	exercise	multiplication_2 exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplication_3	exercises	multiplication_3.html	template	multiplication_3 exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplication_4	exercises	multiplication_4.html	template	multiplication_4 exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplying_decimals	exercises	multiplying_decimals.html	exercise	multiplying_decimals exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit
multiplying_fractions	exercises	multiplying_fractions.html	exercise	multiplying_fractions exercise	2012-08-30	test@example.com	Pending Translation	2012-08-30	edit

Step 2: Tokenize Files and Create Token Records:

This is an admin function that is available only to a restricted number of users on the site.

The process begins with the admin step of tokenizing the original html, js and other files which we generally just refer to as template files. This is done as an admin step. In some cases, this is an automated step where the files are automatically parsed and the tokenization and the entry of the tokens into the database are automated. It is envisioned that there would eventually be a component that would recognize any time a file in the English version was added or changed and these would be entered into a queue for adding to this system. This is only until Khan Academy comes up with a better way to manage this process (hopefully).

Tokenize the Text Content of Template Files

Original html file

```
<div class="info-box hint-box">
    <span class="info-box-header">Need help? Get a hint.</span>
    <div id="get-hint-button-container">
        <input id="hint" type="button" class="simple-button orange full-width" value="I'd like a hint" name="hint"/>
    </div>
    <span id="hint-remainder"></span>
</div>
<div class="info-box related-video-box" style="display:none;">
    <div id="related-video-content">
        <span class="info-box-header">Stuck? Watch a video.</span>
```

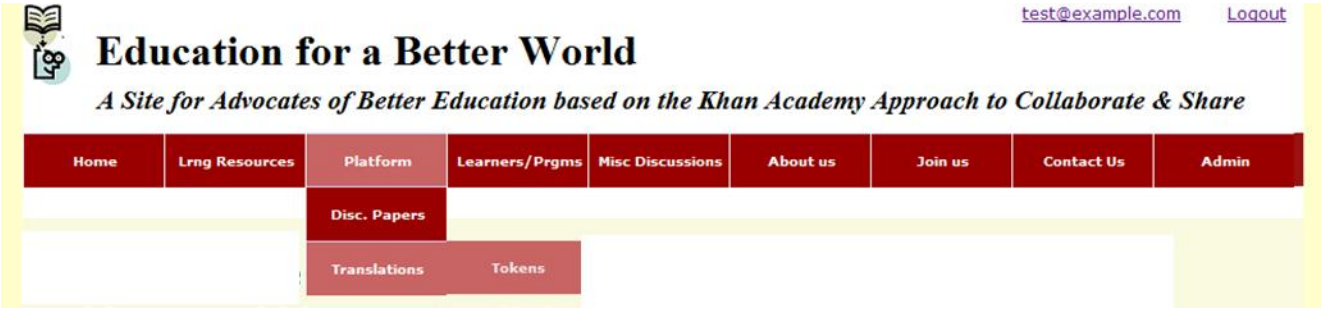
Tokenized html file

```
<div class="info-box hint-box">
    <span class="info-box-header">{{tkn001}}</span>
    <div id="get-hint-button-container">
        <input id="hint" type="button" class="simple-button orange full-width" value="{{tkn002}}" name="hint"/>
    </div>
    <span id="hint-remainder"></span>
</div>
<div class="info-box related-video-box" style="display:none;">
    <div id="related-video-content">
        <span class="info-box-header">{{tkn003}}</span>
```

Token Entries in Database

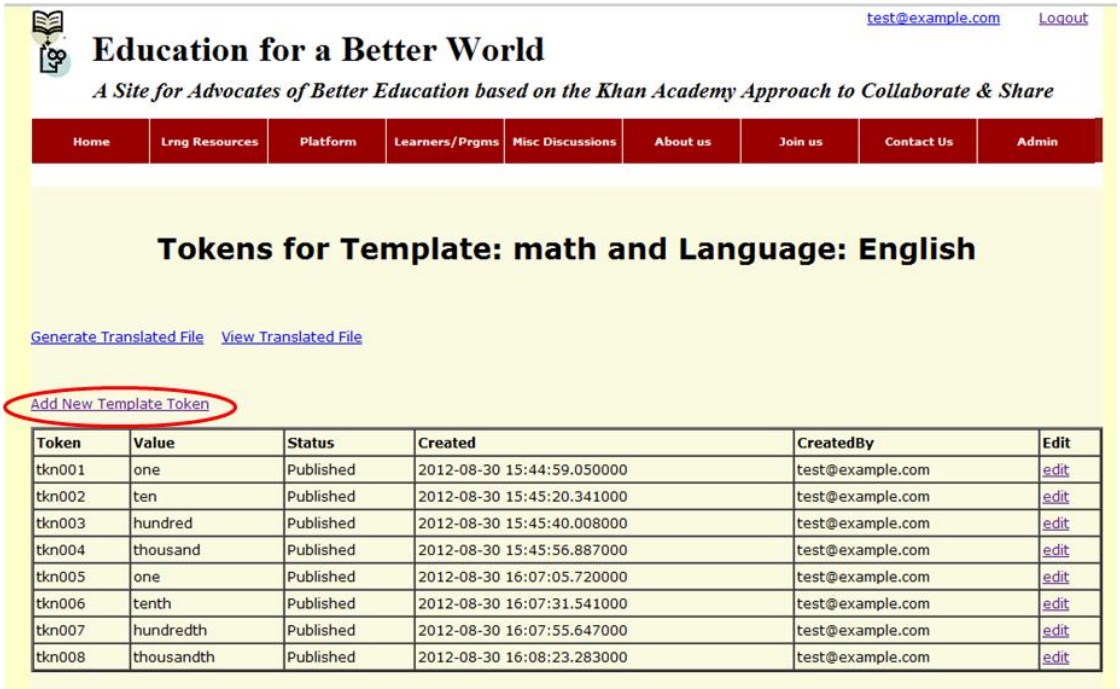
Template	Language	Token	Text
khan-exercise	en	tkn001	Need help? Get a hint.
khan-exercise	en	tkn002	I'd like a hint
khan-exercise	en	tkn003	Stuck? Watch a video.

The Gateway to Entering or Translating Tokens is via the Platform menu.



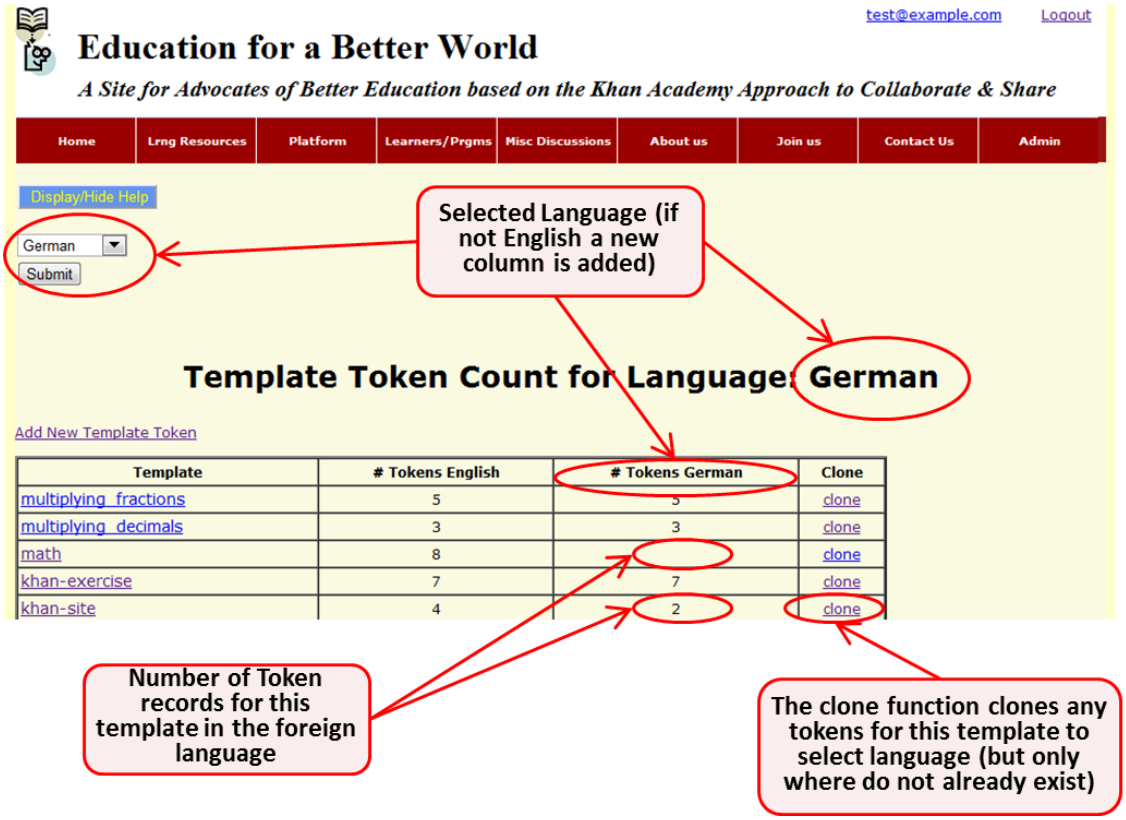
The option to add new Template Tokens is limited to users with certain rights.

Click on the name of the Template to see the Tokens for that template.



Step 3: Cloning Tokens to Another Language:

Tokens are created in English with the values used by Khan Academy. The first step to translating them is to create the records for that language by cloning the English tokens to the target language.



To clone the missing templates (including those that have been added in English since the last time the selected language was cloned), simply click the clone button.

Template Token Count for Language: German

Add New Template Token

Template	# Tokens English	# Tokens German	Clone
multiplying_fractions	5	5	clone
multiplying_decimals	3	3	clone
math	8		clone
khan-exercise	7	7	clone
khan-site	4	2	clone

Before Cloning

Template Token Count for Language: German

Add New Template Token

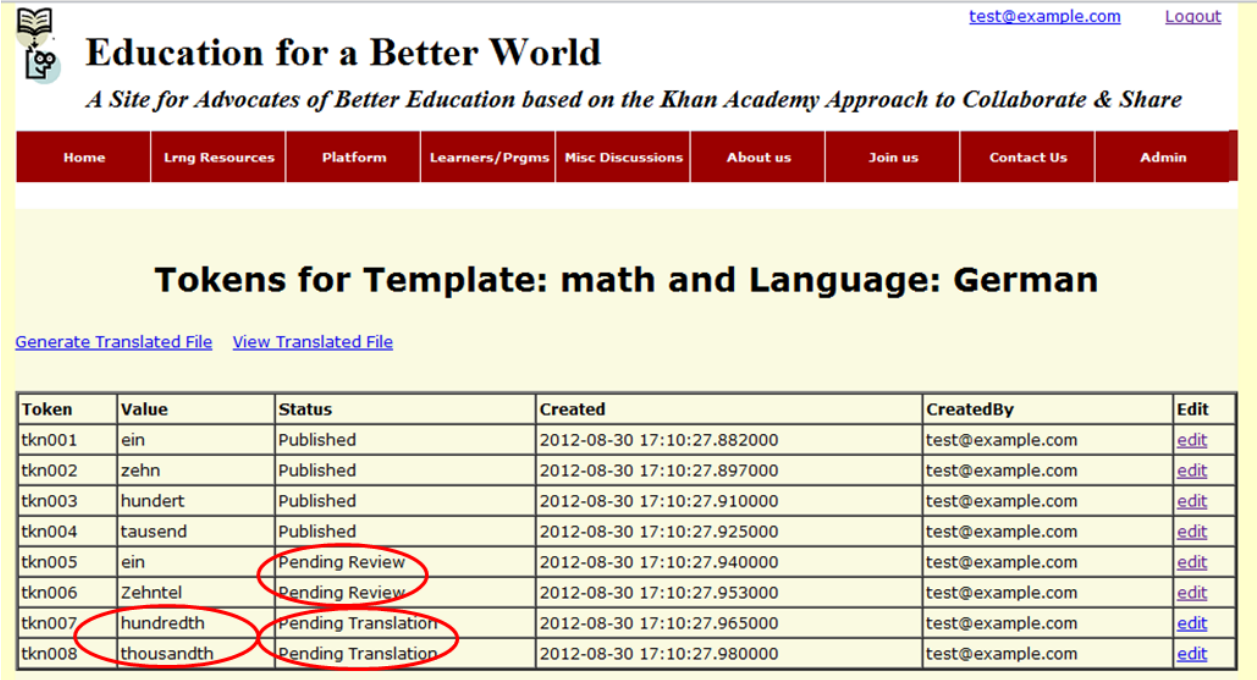
Template	# Tokens English	# Tokens German	Clone
multiplying_fractions	5	5	clone
multiplying_decimals	3	3	clone
math	8	8	clone
khan-exercise	7	7	clone
khan-site	4	4	clone

After Cloning

After cloning, one can see the list of tokens and values for that template file for that language by clicking on the Template Name while in the appropriate language version.

Step 3: Translating Token Values to the Applicable Language:

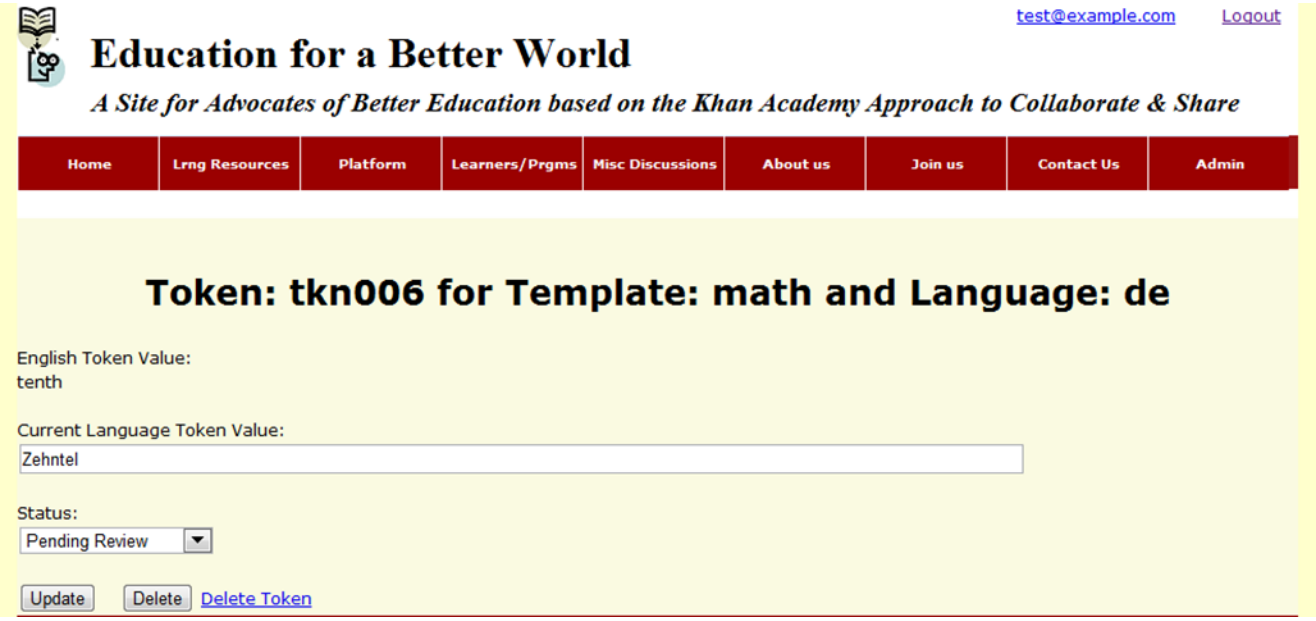
The Token list for a selected Template and Language provides a list of the current values for each Token for that Template and Language as well as its current status (Pending Translation, Pending Review or Published).



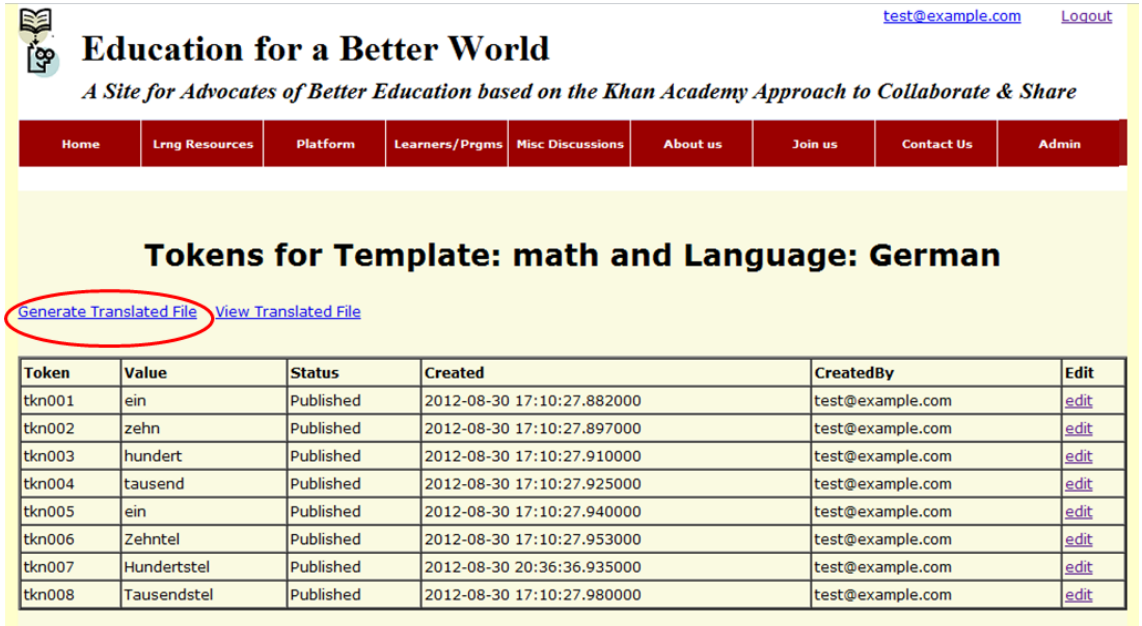
Token	Value	Status	Created	CreatedBy	Edit
tkn001	ein	Published	2012-08-30 17:10:27.882000	test@example.com	edit
tkn002	zehn	Published	2012-08-30 17:10:27.897000	test@example.com	edit
tkn003	hundert	Published	2012-08-30 17:10:27.910000	test@example.com	edit
tkn004	tausend	Published	2012-08-30 17:10:27.925000	test@example.com	edit
tkn005	ein	Pending Review	2012-08-30 17:10:27.940000	test@example.com	edit
tkn006	Zehntel	Pending Review	2012-08-30 17:10:27.953000	test@example.com	edit
tkn007	hundredth	Pending Translation	2012-08-30 17:10:27.965000	test@example.com	edit
tkn008	thousandth	Pending Translation	2012-08-30 17:10:27.980000	test@example.com	edit

Note that the Generate and View Translated Files would not normally be visible until all tokens are published.

To translate or change the status on a token, click on the edit link.



Step 4: Generated Translated Template for the Applicable Language:



test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

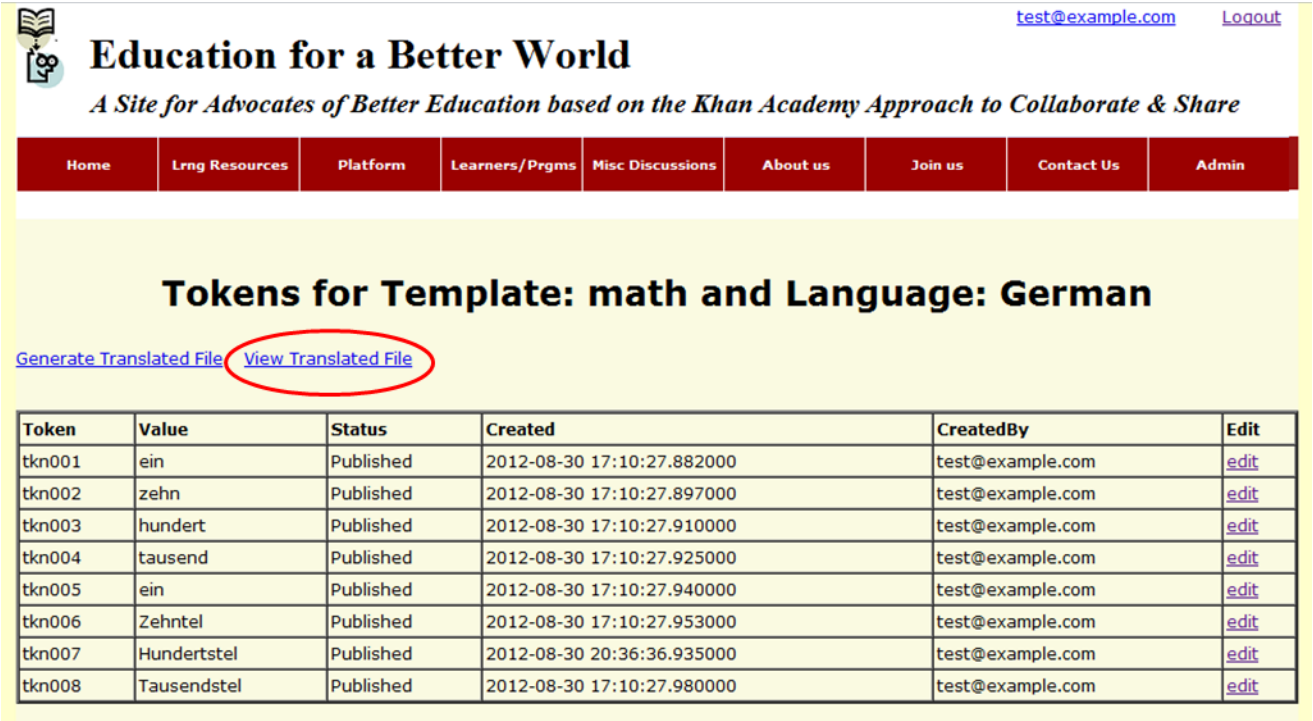
Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

Tokens for Template: math and Language: German

Generate Translated File View Translated File

Token	Value	Status	Created	CreatedBy	Edit
tkn001	ein	Published	2012-08-30 17:10:27.882000	test@example.com	edit
tkn002	zehn	Published	2012-08-30 17:10:27.897000	test@example.com	edit
tkn003	hundert	Published	2012-08-30 17:10:27.910000	test@example.com	edit
tkn004	tausend	Published	2012-08-30 17:10:27.925000	test@example.com	edit
tkn005	ein	Published	2012-08-30 17:10:27.940000	test@example.com	edit
tkn006	Zehntel	Published	2012-08-30 17:10:27.953000	test@example.com	edit
tkn007	Hundertstel	Published	2012-08-30 20:36:36.935000	test@example.com	edit
tkn008	Tausendstel	Published	2012-08-30 17:10:27.980000	test@example.com	edit

Step 5: View Translated Template for the Applicable Language:



test@example.com Logout

Education for a Better World

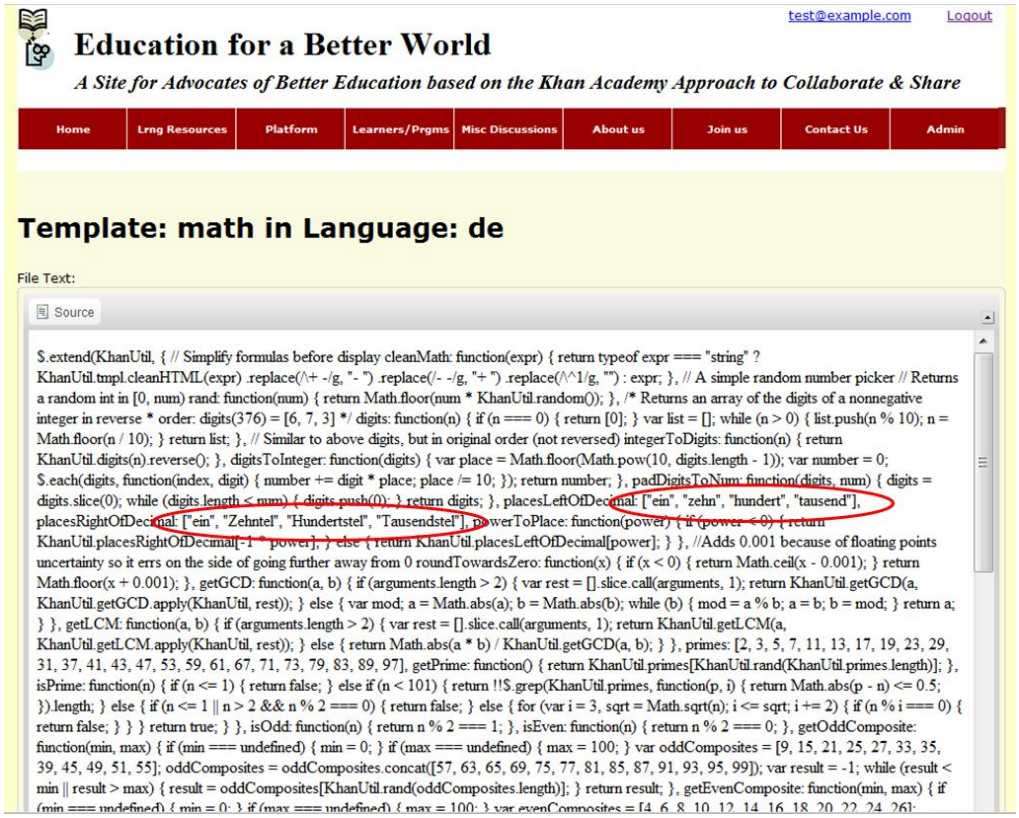
A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

Tokens for Template: math and Language: German

Generate Translated File View Translated File

Token	Value	Status	Created	CreatedBy	Edit
tkn001	ein	Published	2012-08-30 17:10:27.882000	test@example.com	edit
tkn002	zehn	Published	2012-08-30 17:10:27.897000	test@example.com	edit
tkn003	hundert	Published	2012-08-30 17:10:27.910000	test@example.com	edit
tkn004	tausend	Published	2012-08-30 17:10:27.925000	test@example.com	edit
tkn005	ein	Published	2012-08-30 17:10:27.940000	test@example.com	edit
tkn006	Zehntel	Published	2012-08-30 17:10:27.953000	test@example.com	edit
tkn007	Hundertstel	Published	2012-08-30 20:36:36.935000	test@example.com	edit
tkn008	Tausendstel	Published	2012-08-30 17:10:27.980000	test@example.com	edit



test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

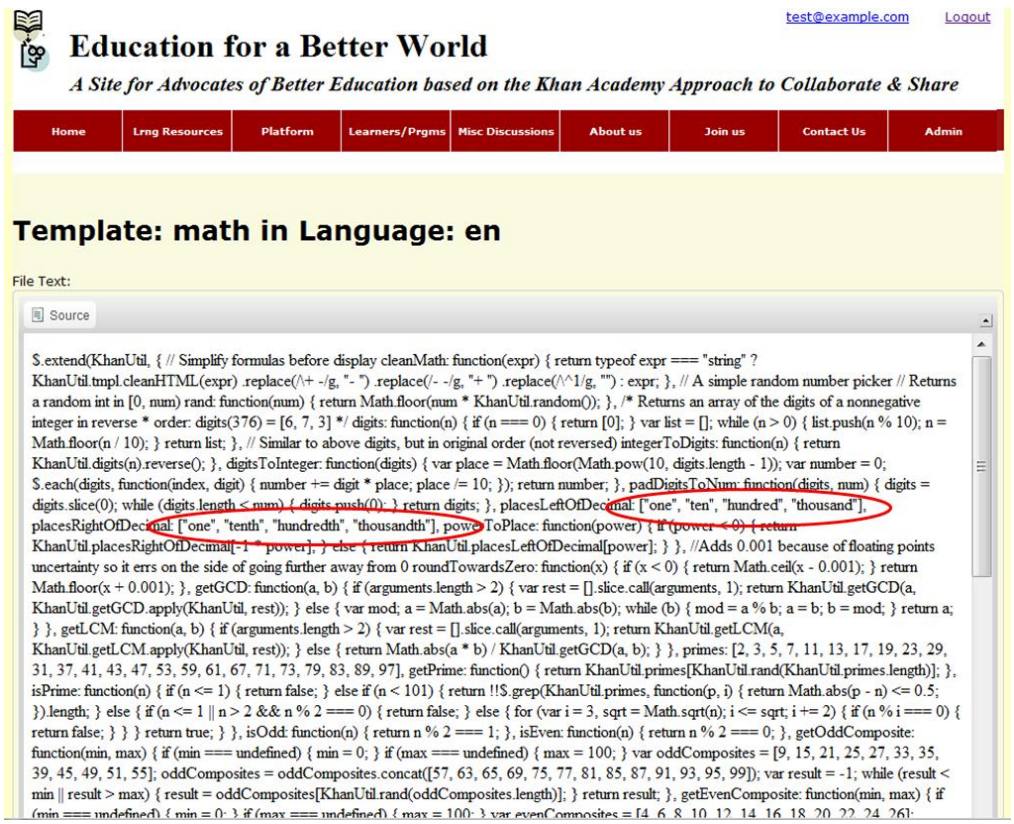
Template: math in Language: de

File Text:

Source

$.extend(KhanUtil, { // Simplify formulas before display cleanMath: function(expr) { return typeof expr === "string" ? KhanUtil.tmpl.cleanHTML(expr) .replace(/\+ -/g, "- ") .replace(/- -/g, "+ ") .replace(/\^1/g, "") : expr; }, // A simple random number picker // Returns a random int in [0, num) rand: function(num) { return Math.floor(num * KhanUtil.random()); }, /* Returns an array of the digits of a nonnegative integer in reverse * order: digits(376) = [6, 7, 3] */ digits: function(n) { if (n === 0) { return [0]; } var list = []; while (n > 0) { list.push(n % 10); n = Math.floor(n / 10); } return list; }, // Similar to above digits, but in original order (not reversed) integerToDigits: function(n) { return KhanUtil.digits(n).reverse(); }, digitsToInteger: function(digits) { var place = Math.floor(Math.pow(10, digits.length - 1)); var number = 0; $.each(digits, function(index, digit) { number += digit * place; place /= 10; }); return number; }, padDigitsToNum: function(digits, num) { digits = digits.slice(0); while (digits.length < num) { digits.push(0); } return digits; }, placesLeftOfDecimal: ["ein", "zehn", "hundert", "tausend"], placesRightOfDecimal: ["ein", "Zehntel", "Hundertstel", "Tausendstel"], powerToPlace: function(power) { if (power < 0) { return KhanUtil.placesRightOfDecimal[-1 * power]; } else { return KhanUtil.placesLeftOfDecimal[power]; } }, //Adds 0.001 because of floating points uncertainty so it errs on the side of going further away from 0 roundTowardsZero: function(x) { if (x < 0) { return Math.ceil(x - 0.001); } return Math.floor(x + 0.001); }, getGCD: function(a, b) { if (arguments.length > 2) { var rest = [].slice.call(arguments, 1); return KhanUtil.getGCD(a, KhanUtil.getGCD.apply(KhanUtil, rest)); } else { var mod; a = Math.abs(a); b = Math.abs(b); while (b) { mod = a % b; a = b; b = mod; } return a; } }, getLCM: function(a, b) { if (arguments.length > 2) { var rest = [].slice.call(arguments, 1); return KhanUtil.getLCM(a, KhanUtil.getLCM.apply(KhanUtil, rest)); } else { return Math.abs(a * b) / KhanUtil.getGCD(a, b); } }, primes: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97], getPrime: function() { return KhanUtil.primes[KhanUtil.rand(KhanUtil.primes.length)]; }, isPrime: function(n) { if (n <= 1) { return false; } else if (n < 101) { return !!$.grep(KhanUtil.primes, function(p, i) { return Math.abs(p - n) <= 0.5; }).length; } else { if (n <= 1 || n > 2 && n % 2 === 0) { return false; } else { for (var i = 3, sqrt = Math.sqrt(n); i <= sqrt; i += 2) { if (n % i === 0) { return false; } } } return true; } }, isOdd: function(n) { return n % 2 === 1; }, isEven: function(n) { return n % 2 === 0; }, getOddComposite: function(min, max) { if (min === undefined) { min = 0; } if (max === undefined) { max = 100; } var oddComposites = [9, 15, 21, 25, 27, 33, 35, 39, 45, 49, 51, 55]; oddComposites = oddComposites.concat([57, 63, 65, 69, 75, 77, 81, 85, 87, 91, 93, 95, 99]); var result = -1; while (result < min || result > max) { result = oddComposites[KhanUtil.rand(oddComposites.length)]; } return result; }, getEvenComposite: function(min, max) { if (min === undefined) { min = 0; } if (max === undefined) { max = 100; } var evenComposites = [4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26];

The same file generated for English (although generation in English is not necessary since that is what we begin with anyway.



test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

Template: math in Language: en

File Text:

Source

$.extend(KhanUtil, { // Simplify formulas before display cleanMath: function(expr) { return typeof expr === "string" ? KhanUtil.tmpl.cleanHTML(expr) .replace(/\+ -/g, "- ") .replace(/- -/g, "+ ") .replace(/\^1/g, "") : expr; }, // A simple random number picker // Returns a random int in [0, num) rand: function(num) { return Math.floor(num * KhanUtil.random()); }, /* Returns an array of the digits of a nonnegative integer in reverse * order: digits(376) = [6, 7, 3] */ digits: function(n) { if (n === 0) { return [0]; } var list = []; while (n > 0) { list.push(n % 10); n = Math.floor(n / 10); } return list; }, // Similar to above digits, but in original order (not reversed) integerToDigits: function(n) { return KhanUtil.digits(n).reverse(); }, digitsToInteger: function(digits) { var place = Math.floor(Math.pow(10, digits.length - 1)); var number = 0; $.each(digits, function(index, digit) { number += digit * place; place /= 10; }); return number; }, padDigitsToNum: function(digits, num) { digits = digits.slice(0); while (digits.length < num) { digits.push(0); } return digits; }, placesLeftOfDecimal: ["one", "ten", "hundred", "thousand"], placesRightOfDecimal: ["one", "tenth", "hundredth", "thousandth"], powerToPlace: function(power) { if (power < 0) { return KhanUtil.placesRightOfDecimal[-1 * power]; } else { return KhanUtil.placesLeftOfDecimal[power]; } }, //Adds 0.001 because of floating points uncertainty so it errs on the side of going further away from 0 roundTowardsZero: function(x) { if (x < 0) { return Math.ceil(x - 0.001); } return Math.floor(x + 0.001); }, getGCD: function(a, b) { if (arguments.length > 2) { var rest = [].slice.call(arguments, 1); return KhanUtil.getGCD(a, KhanUtil.getGCD.apply(KhanUtil, rest)); } else { var mod; a = Math.abs(a); b = Math.abs(b); while (b) { mod = a % b; a = b; b = mod; } return a; } }, getLCM: function(a, b) { if (arguments.length > 2) { var rest = [].slice.call(arguments, 1); return KhanUtil.getLCM(a, KhanUtil.getLCM.apply(KhanUtil, rest)); } else { return Math.abs(a * b) / KhanUtil.getGCD(a, b); } }, primes: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97], getPrime: function() { return KhanUtil.primes[KhanUtil.rand(KhanUtil.primes.length)]; }, isPrime: function(n) { if (n <= 1) { return false; } else if (n < 101) { return !!$.grep(KhanUtil.primes, function(p, i) { return Math.abs(p - n) <= 0.5; }).length; } else { if (n <= 1 || n > 2 && n % 2 === 0) { return false; } else { for (var i = 3, sqrt = Math.sqrt(n); i <= sqrt; i += 2) { if (n % i === 0) { return false; } } } return true; } }, isOdd: function(n) { return n % 2 === 1; }, isEven: function(n) { return n % 2 === 0; }, getOddComposite: function(min, max) { if (min === undefined) { min = 0; } if (max === undefined) { max = 100; } var oddComposites = [9, 15, 21, 25, 27, 33, 35, 39, 45, 49, 51, 55]; oddComposites = oddComposites.concat([57, 63, 65, 69, 75, 77, 81, 85, 87, 91, 93, 95, 99]); var result = -1; while (result < min || result > max) { result = oddComposites[KhanUtil.rand(oddComposites.length)]; } return result; }, getEvenComposite: function(min, max) { if (min === undefined) { min = 0; } if (max === undefined) { max = 100; } var evenComposites = [4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26];

Try Out Translated exercise:

Coming soon. The intention is to provide a version of the Khan Academy Test Program so that the user will be able to see and try the exercise rendered as it would actually appear.

Step 6: Export Translated Site:

Coming soon.

Screen Shots for rounding_numbers Exercise Example:

Round-numbers.html before tokenization:

```
<div data-weight="3">
    <div class="vars">
        <var id="DIGITS">shuffle( [1, 2, 3, 4, 5, 6, 7, 8, 9], 6 )</var>
        <var id="NUM">+DIGITS.join( "" ) / 10000</var>
        <var id="PLACE">randFromArray([ 0, 1, 2 ])</var>
        <var id="TPLACE">placesRightOfDecimal[ PLACE ]</var>
        <var id="ROUNDED">roundTo( PLACE, NUM )</var>
    </div>

    <div class="question">
        <p>Round <code><var>commafy( NUM )</var></code> to the nearest <var>TPLACE</var>.</p>
    </div>
    <div class="solution" data-forms="integer, decimal"><var>ROUNDED</var></div>

    <div class="hints">
        <p>There are two ways to think about this problem.</p>
        <p>1st way: Look at the <var>placesRightOfDecimal[ 1 + PLACE ]</var>s digit <code><var>DIGITS[ 2 + PLACE ]
        </var></code> to see whether to round up or down.</p>
        <p data-if="DIGITS[ 2 + PLACE ] >= 5">Because it is <span data-if="DIGITS[ 2 + PLACE ] > 5">more than
        </span><span data-else>equal to</span> <code>5</code>, we round up, giving <code><var>ROUNDED</var></code>.</p>
        <p data-else>Because it is less than <code>5</code>, we round down, giving <code><var>commafy( ROUNDED )
        </var></code>.</p>
        <p>2nd way:  Consider which end of the number line is closer to <var>NUM</var> - this end is circled in blue.
        </p>
```

Round-numbers.html after tokenization:

```
    <div class="question">
        {{tokenvals.tkn001}}
    </div>
    <div class="solution" data-forms="integer, decimal">
    <div class="hints">
        {{tokenvals.tkn002}}
        {{tokenvals.tkn003}}
        {{tokenvals.tkn004}}
        {{tokenvals.tkn005}}
        {{tokenvals.tkn006}}
```

Tokens for English:

test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

Tokens for Template: rounding_numbers and Language: English

Generate Translated File View Translated File

Add New Template Token

Token	Value	Status	Created	CreatedBy	Edit
TKN001	Round <code><var>commafy(NUM)</var></code> to the nearest <var>TPLACE</var>.	Pending Translation	2012-08-31 01:00:00.583000	test@example.com	edit
tkn002	There are two ways to think about this problem.	Pending Translation	2012-08-31 01:01:58.328000	test@example.com	edit
tkn003	1st way: Look at the <var>placesRightOfDecimal[1 + PLACE]</var>s digit <code> <var>DIGITS[2 + PLACE]</var></code> to see whether to round up or down.	Pending Translation	2012-08-31 01:02:42.758000	test@example.com	edit
tkn004	<p data-if="DIGITS[2 + PLACE] >= 5">Because it is <span data-if="DIGITS[2 + PLACE] > 5">more than<span data-else>equal to <code>5</code>, we round up, giving <code><var>ROUNDED</var></code>.	Pending Translation	2012-08-31 01:03:20.532000	test@example.com	edit
tkn005	<p data-else>Because it is less than <code>5</code>, we round down, giving <code> <var>commafy(ROUNDED)</var></code>.	Pending Translation	2012-08-31 01:04:10.792000	test@example.com	edit
tkn006	p>2nd way: Consider which end of the number line is closer to <var>NUM</var> - this end is circled in blue.	Pending Translation	2012-08-31 01:04:39.545000	test@example.com	edit

Tokens in German:

test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

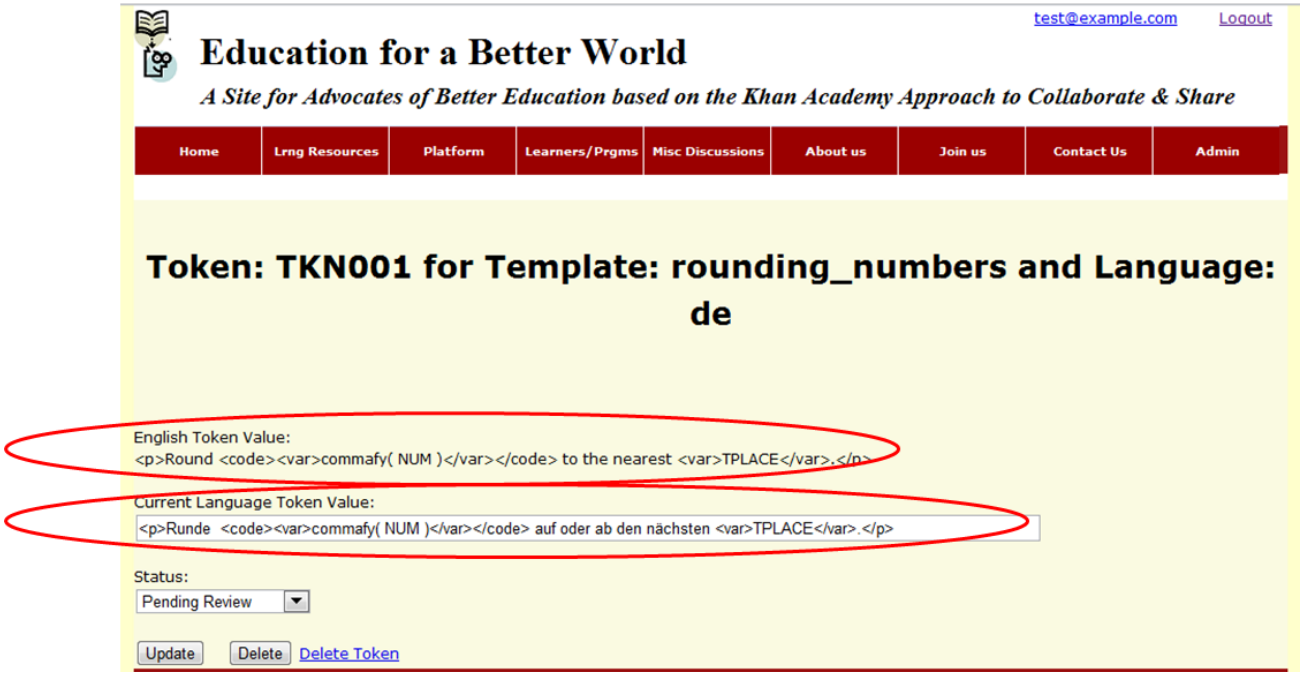
Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

Tokens for Template: rounding_numbers and Language: German

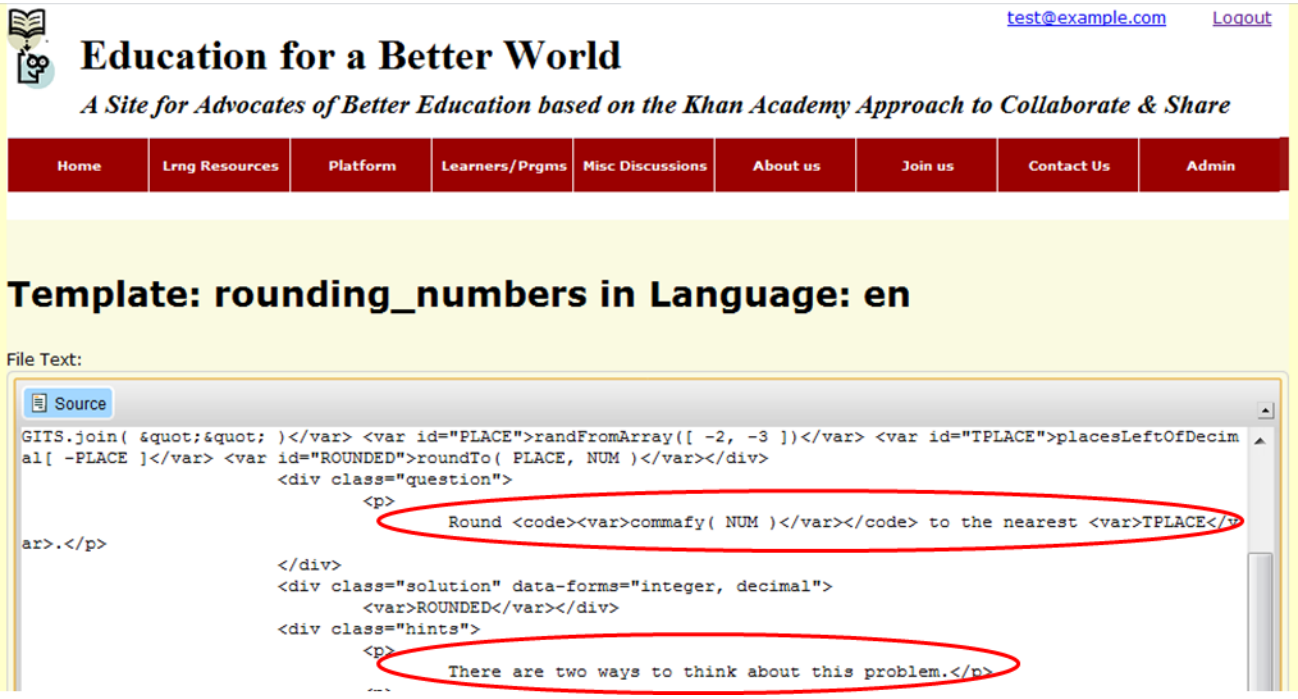
Generate Translated File View Translated File

Token	Value	Status	Created	CreatedBy	Edit
tkn001	Runde <code><var>commafy(NUM)</var></code> auf oder ab zum naechsten <var>TPLACE</var>.	Pending Translation	2012-08-31 01:33:32.463000	test@example.com	edit
tkn002	Es gibt zwei Wege dieses Problem anzugehen.	Pending Translation	2012-08-31 01:08:58.494000	test@example.com	edit

Token Translation Screen:



Generated Exercise File in English:



Generated Exercise File in German:

test@example.com Logout

Education for a Better World

A Site for Advocates of Better Education based on the Khan Academy Approach to Collaborate & Share

Home | Lrng Resources | Platform | Learners/Prgms | Misc Discussions | About us | Join us | Contact Us | Admin

Template: rounding_numbers in Language: de

File Text:

Source

```
<p>
        <meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />
</p>
<p>
        <title></title>
        <script src="../khan-exercise.js"></script></p>
<div class="exercise">
        <div class="problems">
                <div data-weight="2">
                        <div class="vars">
                                <var id="DIGITS">shuffle( [1, 2, 3, 4, 5, 6, 7, 8, 9], 5 )</var> <var id="NUM">+DI
GITS.join( &quot;&quot; )</var> <var id="PLACE">randFromArray([ -2, -3 ])</var> <var id="TPLACE">placesLeftOfDecim
al[ -PLACE ]</var> <var id="ROUNDED">roundTo( PLACE, NUM )</var></div>
                        <div class="question">
                                <p>
                                        Runde <code><var>commafy( NUM )</var></code> auf oder ab zum naechsten <va
r>TPLACE</var>.</p>
                        </div>
                        <div class="solution" data-forms="integer, decimal">
                                <var>ROUNDED</var></div>
                        <div class="hints">
                                <p>
                                        Es gibt zwei Wege dieses Problem anzugehen.</p>
```