JUICE site structure and customization overview

This document describes the components of the JUICE site, the hierarchy of the underlying JUICE data in Amazon S3, and provides some general instructions about how to customize JUICE.   
  
It assumes that you have already instantiated the JUICE application and followed instructions to update the JUICE default .config file with the appropriate URLs and codes. It also assumes that you have set up S3 with 2 buckets:

Main JUICE content bucket

JUICE narration bucket

Note: If you are not using S3, most of this information still applies to the directories you have set up for the JUICE FILES.

Detailed information about customizing JUICE can be found in separate documents that are listed in the Customizable components of JUICE and available documentation section.

Contents

[About the JUICE Site 2](#_Toc523475634)

[Basic site components—modules and modulettes (mini-lessons) 2](#_Toc523475635)

[Additional site components 4](#_Toc523475636)

[Amazon S3 set up 4](#_Toc523475637)

[Main JUICE Bucket structure 5](#_Toc523475638)

[Alternate JUICE program support 9](#_Toc523475639)

[JUICE narration support 11](#_Toc523475640)

[Links to external files 12](#_Toc523475641)

[How JUICE loads pages—JUICE URLs and S3 directories 13](#_Toc523475642)

[Customizing JUICE 14](#_Toc523475643)

[Site text and UI 14](#_Toc523475644)

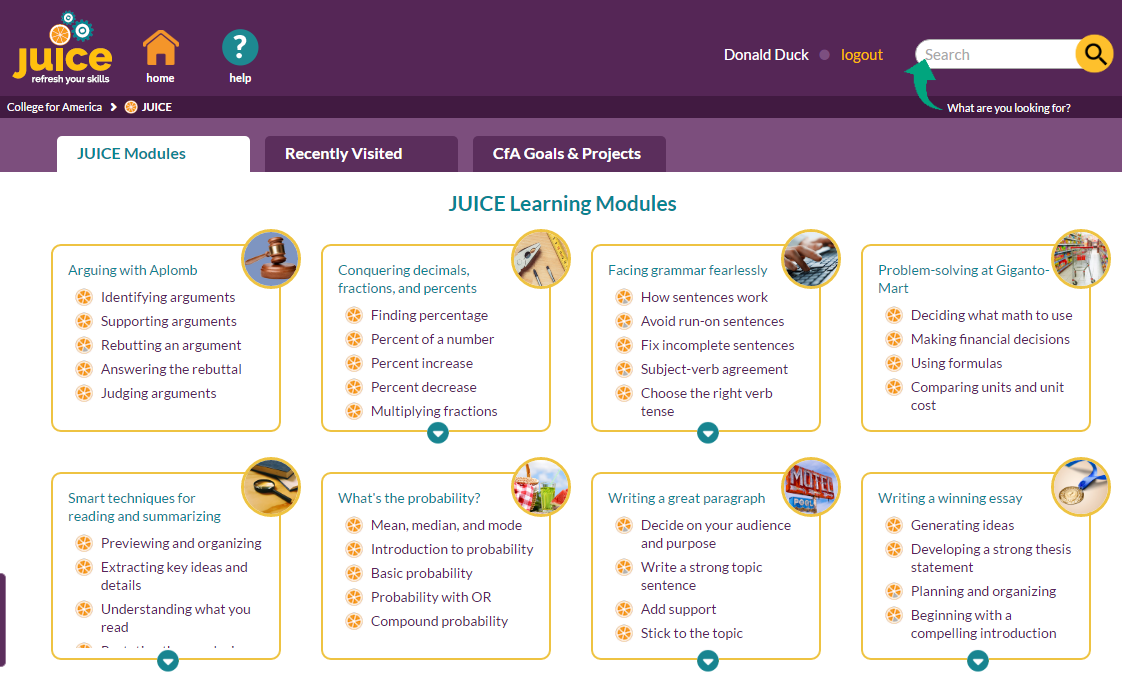
[Content files 15](#_Toc523475645)

[Customizable components of JUICE and available documentation 16](#_Toc523475646)

# About the JUICE Site

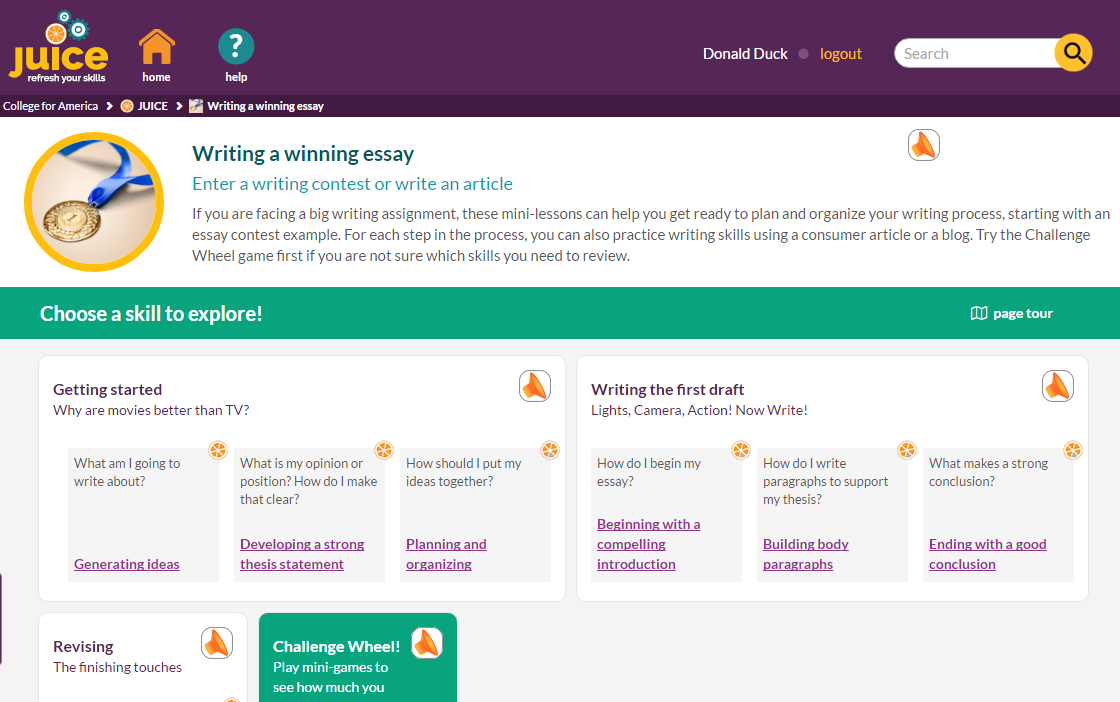
## Basic site components—modules and modulettes (mini-lessons)

The building block for the JUICE site is the **module**:



modules

Each module contains one or more “modulette groups” (mini-lesson groups), and each modulette group contains one or more modulettes (mini-lessons):

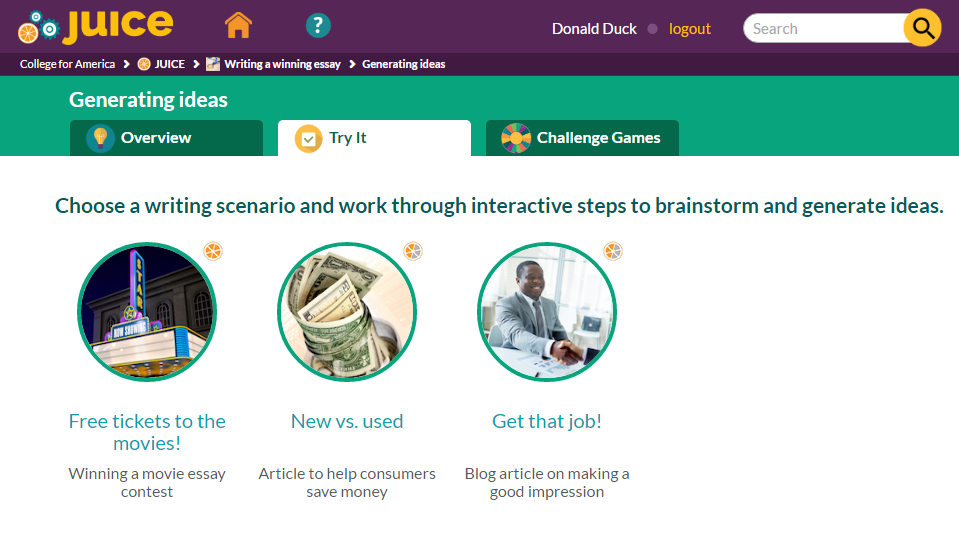


A module may also include a special game called a “Challenge Wheel.” This is a summary game designed for low stakes assessment of the skills covered in the module mini-lessons.

modulette groups

modulettes (mini-lessons)

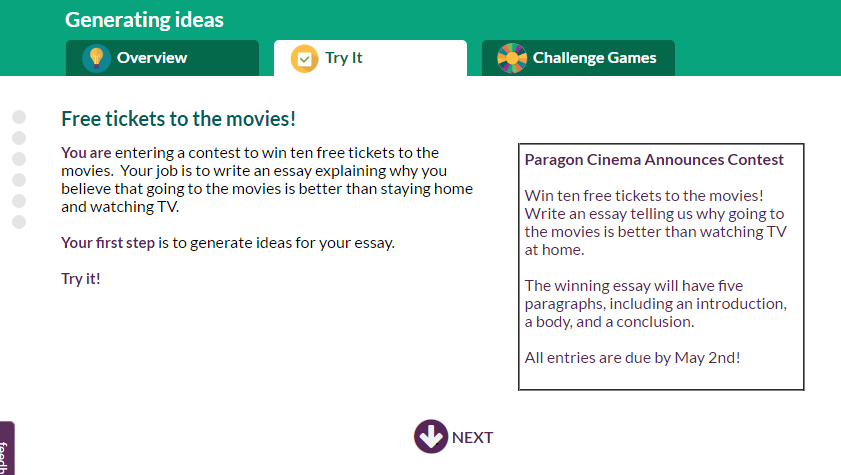
Each modulette is configured to have multiple parts (tabs). Each tab can have more than one choice (track). Each track has a content player (widget)—a track and widget together are called a widget instance. Each widget instance has a content file. Together, the widget instance and the content file deliver the learning experience.



Track + widget = widget instance

tracks

tabs



widget instance + file = student experience

## Additional site components

The site has additional components that make up the complete user experience.

**Audio:** Modules and modulettes both play **audio narration files**. Modulettes may also make use of an **audio map** to override default audio narration files.

**External files:** Modulettes can link to **external files** (ancillary files) that contain content that doesn’t fit within the framework of the mini-lesson. These can include images or PDF documents that contribute to the content of the mini-lesson.

**Photos and images:** Modules, Try its, and Challenge Games use photos or graphic images as identifiers in the site.

**Projects:** The Goals and Projects view of the JUICE Dashboard shows a list of **projects** grouped by goals—and the modules that support each project.

**Rubric criteria:** Both modules and modulettes can be mapped to **rubric criteria**. The rubric criteria are used by Search.

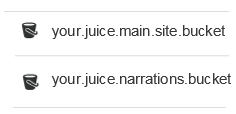
**Competencies and sub competencies:** Both modules and modulettes can be mapped to competencies and subcompetencies. These are save in the data and used only for data analysis.

# Amazon S3 set up

The JUICE web application uses json formatted files to control the text and content displayed in the JUICE user interface. These files are stored in S3, along with the site’s external files, photos and images, and audio narration files.

For most of the site content, JUICE is designed to support several S3 instances, such as dev, qa, and prod. The server that JUICE is running on pulls from its associated S3 bucket.

However, JUICE supports only one instance of the narration audio files. These files are in a separate narration bucket and all servers pull from this bucket.  
  
The most basic set up of JUICE requires two buckets with names that are defined in config.js. You can use any names for the buckets. The names used in these screen shots are placeholder examples:



Main JUICE bucket—prod instance

JUICE audio files bucket—used for all instances

your.main.juice.content.bucket

## Main JUICE Bucket structure

The directory structure of the main S3 bucket reflects the hierarchy of the site. The directory names and file names in the bucket are used as the unique identifiers for the contents of the directory.

This is the directory structure for the Essays module shown in the screen shots above.



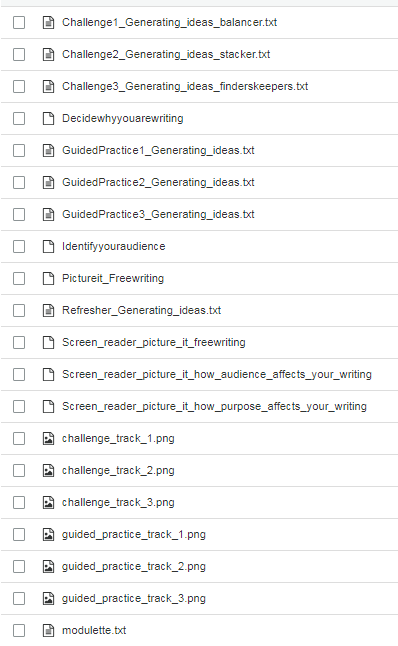
Bucket for JUICE production instance



Directory for module CS17.essays



Directory for modulette CS17.1.Generating ideas

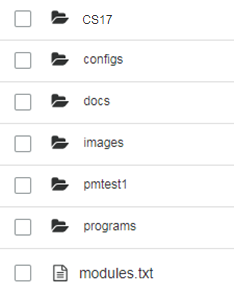


Note: S3 interprets a “.” in a file name as a slash (/) symbol representing levels in the S3 directory structure.

Full JUICE file name: CS17.1.Refresher\_Generating\_ideas.txt

Files for modulette CS17.1.Generating ideas

The **top level** of the JUICE content bucket includes the module directories, a few resource directories, and the file **modules.txt**, which defines the contents of the core JUICE program:

Module directory  
  
Special configuration files  
  
Files for the help system, license agreements, and privacy policies, and other documents and files that are shared across the site.

Image files that are shared across the site  
  
Testing and sandbox files  
  
Directories and information for alternate JUICE programs, see below

JUICE program description file, defines the contents of the JUICE program

Each **module directory** includes modulette directories, files for the challenge wheel game, the module image file, and **module.txt**, which defines the contents of the module:



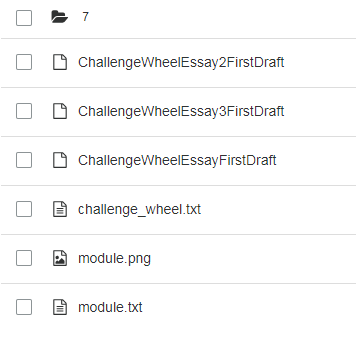
1

Modulette directory  
  
  
  
External content files for the Challenge Wheel game

Challenge Wheel widget file

Module image file

Module description file, defines the contents of the module



Each **modulette** directory contains the widget, image, and ancillary files needed to display the contents of the mini-lesson, and **modulettes.txt,** which defines the contents of the modulette.





Challenge game widget files

External content file  
  
Try it widget files

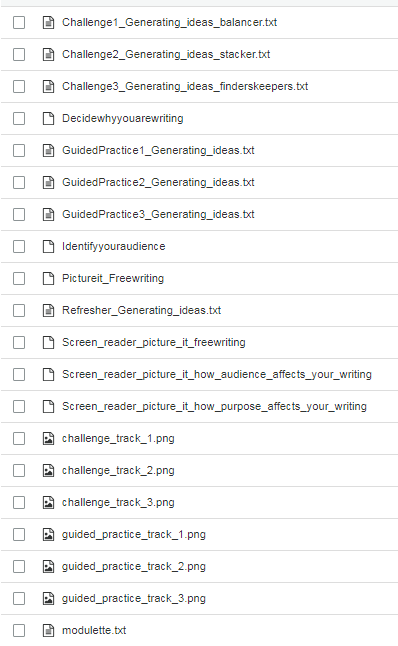
External content files  
  
Overview widget file

External content files

Challenge-game choose page images

Try it choose page images

modulette description file, defines the contents of the modulette



## Alternate JUICE program support

The main JUICE program is called “core JUICE” in this documentation. It consists of 11 modules and user interface text that was designed for the original College for America program that JUICE was designed to support. Core JUICE can be modified for your program. Review the documentation about customizing JUICE to learn more.

JUICE also allows the creation of alternate programs that can be customized in several ways. You can

1. re-order or choose a subset of modules to include in the program, but re-use the content and the site UI from the core JUICE.
2. customize some or all the existing content or add new content.
3. customize the program’s site UI.

See the documentation on creating alternate programs for more information.

The files needed for each alternate program are in the **programs** directory. The **programs** directory includes a directory for each alternate program, and the file **program\_id\_mapping.json.** This defines the program keys and ids that are required to access each program, and other parameters that are stored with the user and activity data:







Alternate program directories

Program id mapping file. Defines program ids and other parameters that are stored with user and activity data.

The directory structure within each alternate program directory repeats the structure of the core JUICE program, starting with a set of module directories and files at the top level of the alternate program’s directory.

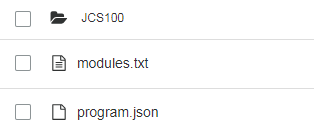
The top level of an alternate program’s directory also contains an additional **program.json** file, which defines the customizable aspects of the user interface:

Program module directory

Program description file, defines the contents of the program

Program customization file, defines the customizable aspects of the user interface.



Note: If an alternate program does not include any custom content, the module and modulette directory structure will be in place, but there will be no widget, image, or content files.

The only files included will be the json files that define the program contents, the program customizations, the module contents, and the modulette contents:

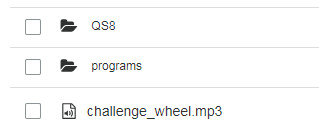
modules.txt  
 program.json  
 module.txt  
 modulette.txt

## JUICE narration support

There is only one bucket for JUICE narration. It has the same organization and hierarchy as the S3 content bucket.

The top level contains directories for each module, a programs directory for alternate JUICE programs, and the audio file that is used for all the challenge wheel narrations:

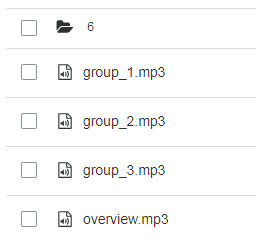




Module directory  
  
Alternate programs directory  
  
Narration for challenge wheel game, used in all modules

The module directories contain the narration that is used for the module page, and directories for each modulette:





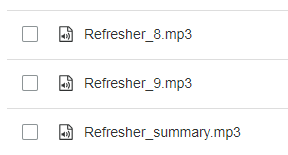
Modulette directory

Audio files for module groups

Audio file for module introduction

The modulette directories contain the narration files for each of the refresher steps. Some modulette directories also contain an audio mapping file that is used to overwrite the automatic name-based association of refresher steps and audio files. See the document on customizing and adding narration for more information.



Refresher step audio files

Audio mapping file (optional)



## Links to external files

Some files contain links to external files that are stored in the modulette or module directories of S3. These links use variables to represent the path of the bucket that the file is stored in:

S3\_BUCKET\_/[bucket directory/file name]

{{fileResources}}[bucket directory/file name] (the missing slash symbol is intentional!)

S3\_FILE\_PATH/[bucket directory/file name]

JUICE will swap the highlighted text for the URL and name of your S3 bucket, based on the values you define in the site config files.

These variables are supported in the following locations:

|  |  |  |
| --- | --- | --- |
| **Variable** | **What type of file can use it** | **Name of file** |
| S3\_BUCKET\_ | Refresher, Guided Practice, and Challenge Game files (widget content files)  Module description  Modulette tab description | Individual widget files  module.txt  modulette.txt |
| {{fileResources}} | JUICE help | /docs/help/help.htm |
| S3\_FILE\_PATH | Program definition—used to define paths to the help documents | /components/program-value/program-value-defaults.json  program.json |

# How JUICE loads pages—JUICE URLs and S3 directories

The JUICE site loads pages based on a URL that reflects the page being loaded. For example, a URL that includes this:

/juice/dashboard/dashboardmodules

loads the JUICE Dashboard showing the JUICE Modules tab.

URLs that are loading modules or modulettes include information about the S3 directory in which the the module or modulette files are stored.

JUICE module page URLS includes this:

/juice/module/[moduleid]

Example:

/juice/module/CS28CCT3.arguments

The part of the URL that is after **module/** is the S3 directory for the module. The second part of the module id is not needed to load the correct module:



Not needed by the site

/juice/module/CS28CCT3.arguments

JUICE modulette page URLS include this:

/juice/modulette/[module id].[modulette id].[modulette name]

Example:

/juice/modulette/CS28CCT3.1.Identifying%20arguments

The part of the URL that is after **modulette/** is the S3 directory for the modulette. The modulette name is not needed to load the correct modulette, but the period before the modulette name is!



/juice/modulette/CS28CCT3.1.Identifying%20arguments



This URL will load the same modulette:

/juice/modulette/CS28CCT3.1. The ending period is required!

# Customizing JUICE

## Site text and UI

You can customize almost any element of the JUICE site UI by editing the correct json file or .txt file as follows:

1. Download the file from S3
2. Open it in Brackets or some other structured text editor
3. Make the changes you desire
4. Re-upload it to S3
5. Clear your cache to see the changes

The documentation listed in the table below explains what you can change and how the different parts of the json and txt files affect the site.

Tip: Set S3 to support versioning to protect yourself against any mistakes.

## Content files

If you have upload the JUICE content to Amazon S3, you can customize Refresher (Overview), Guided Practice (Try it), and Challenge (Challenge games) files by opening them in the correct widget file authoring tool, making changes, and saving. You can preview your changes in a similar manner.  
  
Note: Your AWS key and secret must be set in the site code’s local.json file for you to use these editing tools.

See the documentation on content authoring for information about how to use the authoring tools, and for the language and formatting conventions used in the site.

To access the content authoring and previewing tools:

1. Access the test log-in page for your instance of JUICE at /juice/oauth\_test
2. Choose a user with a role of Admin
3. Open the correct file authoring tool by putting this URL into your browser:

/juice/author/[widget type]/[module].[modulette].[S3 filename]

Example:

/juice/author/refresher\_widget/CS17.1.Refresher\_Generating\_ideas

1. To **preview** your changes, first **SAVE** them. The file will be saved to the S3 bucket that you have defined as your dev bucket in juice.js.
2. Use this URL to preview the file:

/juice/preview/[widget type]/[module].[modulette].[S3 filename]

Example:

/juice/preview/refresher\_widget/CS17.1.Refresher\_Generating\_ideas

IMPORTANT! The widget type in the URL MUST specify the correct widget! Otherwise, you can damage your file. There is NO PROTECTION against this!

To decide what widget type to use in the URL, look at the file name and then choose the correct widget type from the table below.

If the file is a challenge wheel file, the widget type will not be listed. In this case, play the game in the site to see which type of game it is.

Any other game file without a widget type is a test file, and should be removed from your S3 instance.

**Widget type table**

| **If the filename includes this:** | **The widget type is this:** |
| --- | --- |
| Refresher | refresher\_widget |
| GuidedPractice | guided\_practice\_widget |
| balancer | balancer\_widget |
| quickpick | quick\_pick\_widget |
| sorter | sorter\_widget |
| puzzler | puzzler\_widget |
| fridgemagnets | fridge\_magnets\_widget |
| finderskeepers | finders\_keepers\_widget |
| pickandstack | pick\_and\_stack\_widget |
| stacker | stacker\_widget |
| perfectword | perfect\_word\_widget |
| highlighter or spotlight | highlighter\_widget |
| fixit | fix\_it\_widget |

## Customizable components of JUICE and available documentation

The documents referenced here are included in the JUICE documentation directory.

| **Component** | **What can be customized** | **S3 file to edit** | **Documentation name** |
| --- | --- | --- | --- |
| Basic UI of “core JUICE” | * Site title and logo * Privacy agreement requirement * Help * Breadcrumbs * Availability of tabs on JUICE home * UI text for JUICE home tabs * Welcome popup “learn more” availability and text * Availability of audio narration | This file is NOT in S3.    /components/program-value/program-value-defaults.json | Customizing basic UI components and creating alternate programs |
| JUICE Home—Modules Library and Recently Visited tabs | * Modules and modulettes included in the program * Other attributes of the modules and modulettes, such as names and ordering. | modules.txt | Customizing the JUICE Dashboard |
| JUICE Home—Goals and Projects tab | * Goal and Project names * Module mapping   Note: This view can hidden or further customized. See “Alternate program UI” for more information | modules.txt | Customizing the JUICE Dashboard |
| Module page and structure | * Welcome pop-up text * Mini-lessons included in each module * User interface text that appears on the module page in the site * Additional search criteria—tags and rubrics * Goals and projects page correlations * Competency correlations | module.txt | Customizing a JUICE Module |
| Modulette (mini-lesson) structure | * Parts, choices, content players for each mini-lesson * File names for Overview, Try it and Challenge game content files * User interface text that appears on the mini-lesson pages in the site * Additional search criteria—tags and rubrics * Sub competency correlations * Mappings for audio files | modulette.txt | Customizing a JUICE mini-lesson |
| Mini-lesson content | * Overviews, Try its, and mini-games | Misc widget files | Customizing and creating new mini-lesson files  Various documentation files in the JUICE authoring directory |
| External files | * External file names and management tools | Misc external files | Adding and editing Picture its and other external files |
| Narration | * When narration plays and when it doesn’t * Specific narration segments | Misc narration files  audio-manifest.txt | Adding and customizing narration |
| Search tags | * Tags and rubrics associated with modules, modulettes, and external files. | This is NOT in S3:  Database table | Using tags to improve JUICE search |
| Alternate program UI | * Site title and logo * Privacy agreement requirement * Help * Breadcrumbs * Availability of tabs on JUICE home * UI text for JUICE home tabs * Welcome popup “learn more” availability and text * Availability of audio narration | program.json | Customizing basic UI components and creating alternate programs |
| Alternate program access | * Program id. * Program key used to access the program * Parameters that are stored with user and activity records | program\_id\_mapping | Customizing basic UI components and creating alternate programs |