

Overview

This document is meant to guide a user through Reactome's Release. It will primarily cover running Release through Jenkins but will [eventually] also contain information on specific steps, FAQs, common errors, etc. It is meant to allow anyone to run a Reactome Release with minimal prior knowledge.

At a high level, Reactome's Release is the process of taking 'approved' annotations in the curator MySQL database (*gk_central*) into a release MySQL database (*test_slice*, which gets turned into the actual release database *release_current*) updating/processing this database with Release-specific scripts, converting the relational database into a final graph database, generating data files from both databases, and then finally releasing all of this onto the live site, <https://reactome.org>.

To simplify all of this, we use Jenkins. Jenkins is a tool commonly used for CI/CD applications, though Reactome uses it more as a glorified script runner. Each 'step' that needs to be run during Release typically involves a number of command-line actions (taking database backups, building and running the program, emailing, archiving everything and cleaning up) that would generally require an experienced Reactome developer. Jenkins attempts to simplify the execution of these actions, leaving the developer mostly to the task of validation.

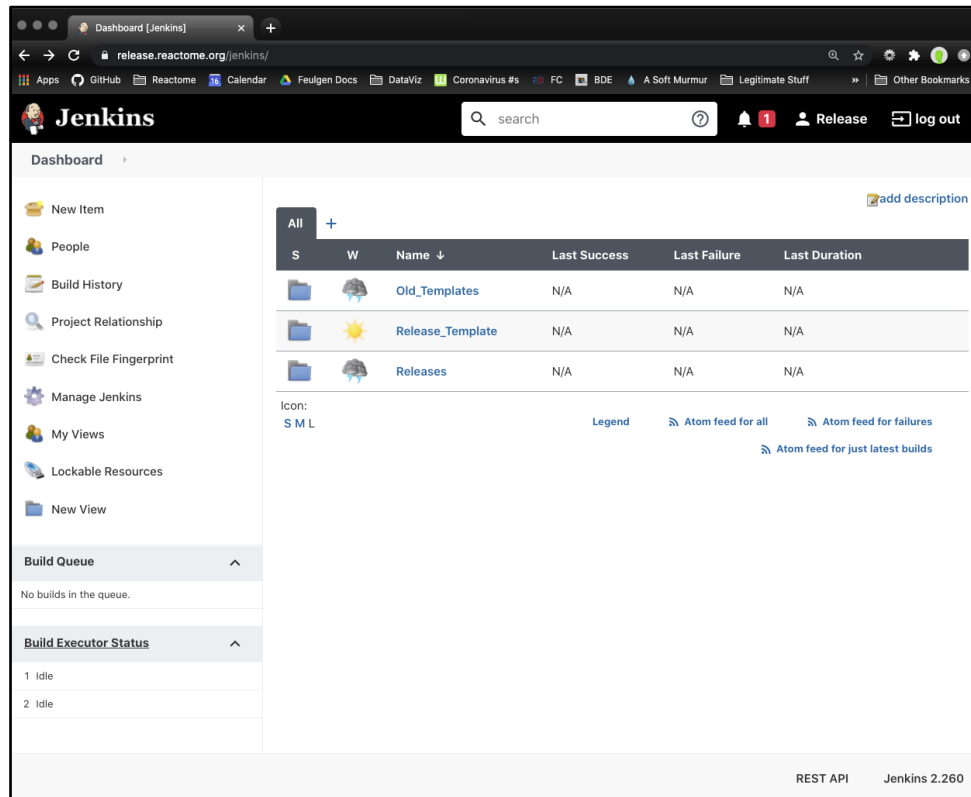
Broadly speaking, there are 7 'phases' of Release in Jenkins:

- **Setup:** Preparing the Jenkins Release environment.
- **Pre-slice:** This describes steps that update the 'gk_central' database before the 'release' (*slice_test*) database has been created.
- **Relational-Database-Updates:** Steps that directly update the relational database.
- **GenerateGraphDatabaseAndAnalysisCore:** A single step where the relational database is imported to a graph database. The analysis core is produced as well.
- **File-Generation:** Generates files from the relational and graph databases. These files are generally for user downloads and images/diagrams displayed on the website.
- **SearchIndexer:** A single step that creates the search indexer used on the website.
- **Post-Release:** Steps to be run after the Release has gone live on the website.

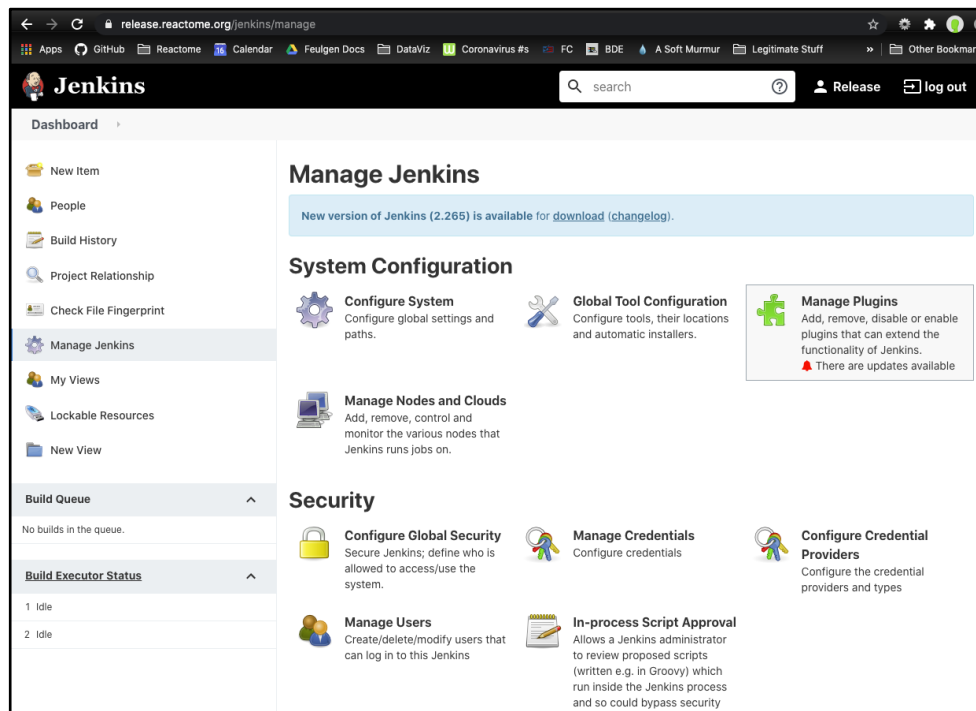
Setup

This section describes setting up a fresh Release environment on the Release Jenkins server, found at <https://release.reactome.org/jenkins/>. You will need the **Jenkins credentials** to log on. If you do not have these credentials, ask a fellow developer or your manager.

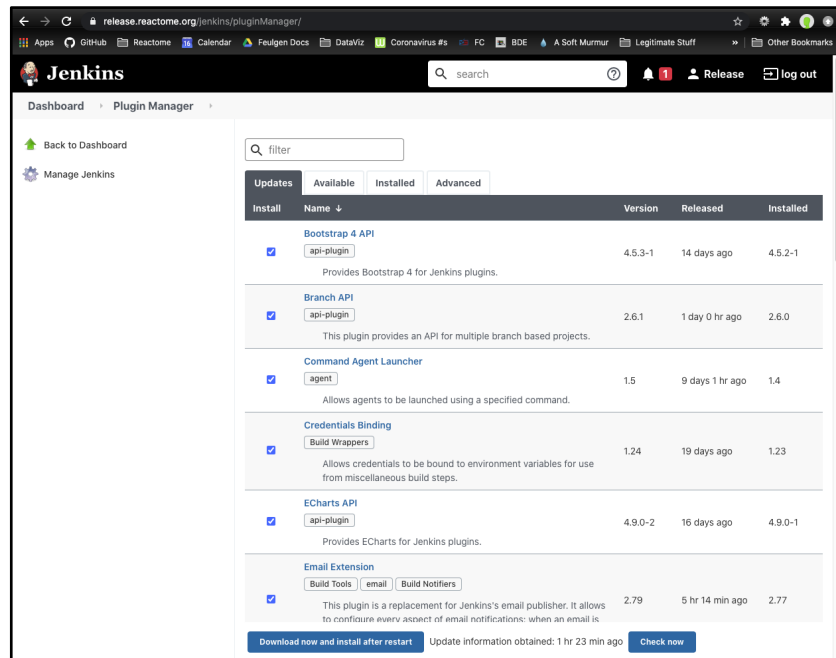
When you first log on, you will be at the Jenkins *Dashboard*.



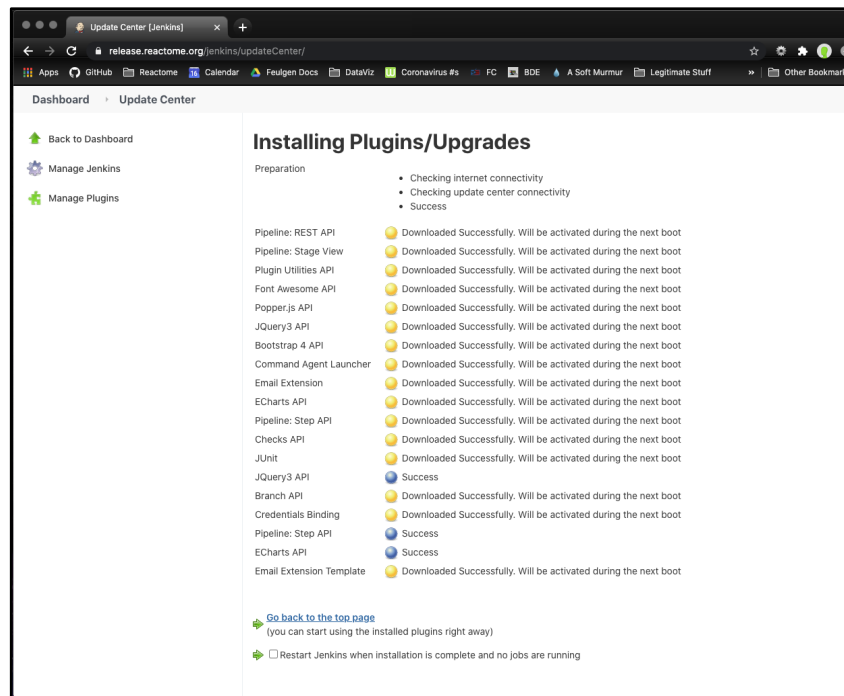
First, we will want to update any packages used by Jenkins, to make sure they are all up to date. Select *Manage Jenkins* on the left-hand side, and then *Manage Plugins* on the next page.



This will bring you to the *Plugin Manager*. This page has tables that contain lists of available updates to installed packages, available packages, and packages currently installed. If there are any available updates, select the ones you wish to install (likely all of them) and then click “*Download now and install after restart*” at the bottom of the window.



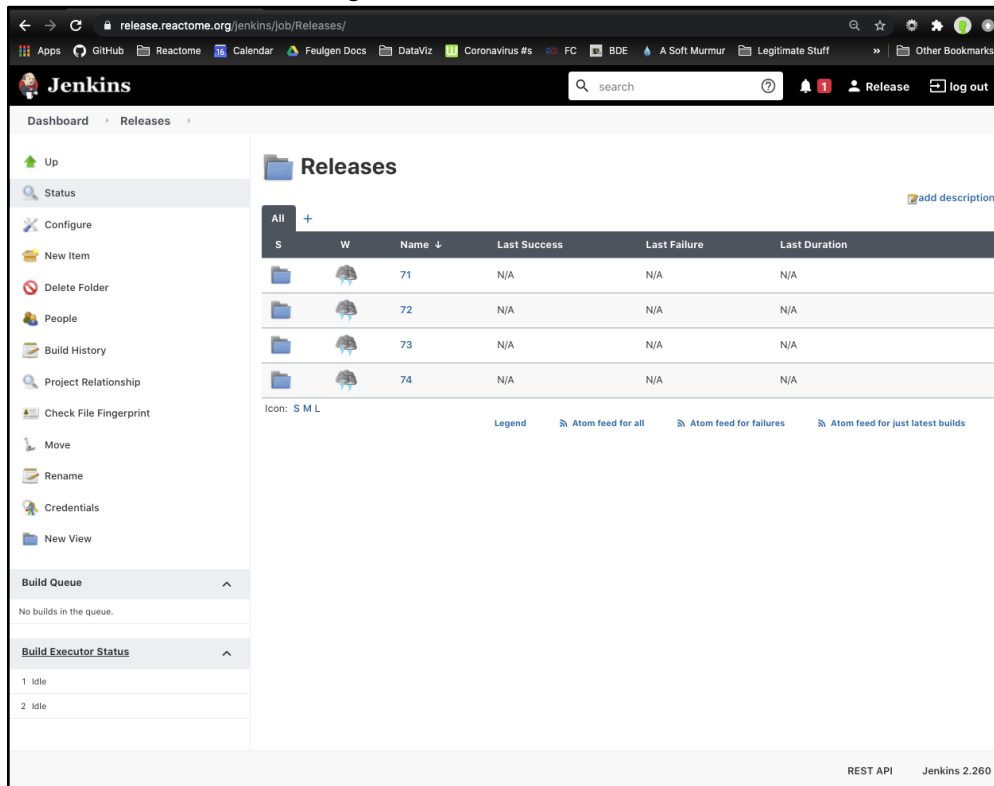
This will bring up a downloads window for each of the updates you selected. Select *Restart Jenkins when installation is complete and no jobs are running* to automate the required restart.



Once all updates have been installed and Jenkins restarts, you will need to log back in.

Next, we will be creating a new Release folder by copying the contents of the *Release_Template*.

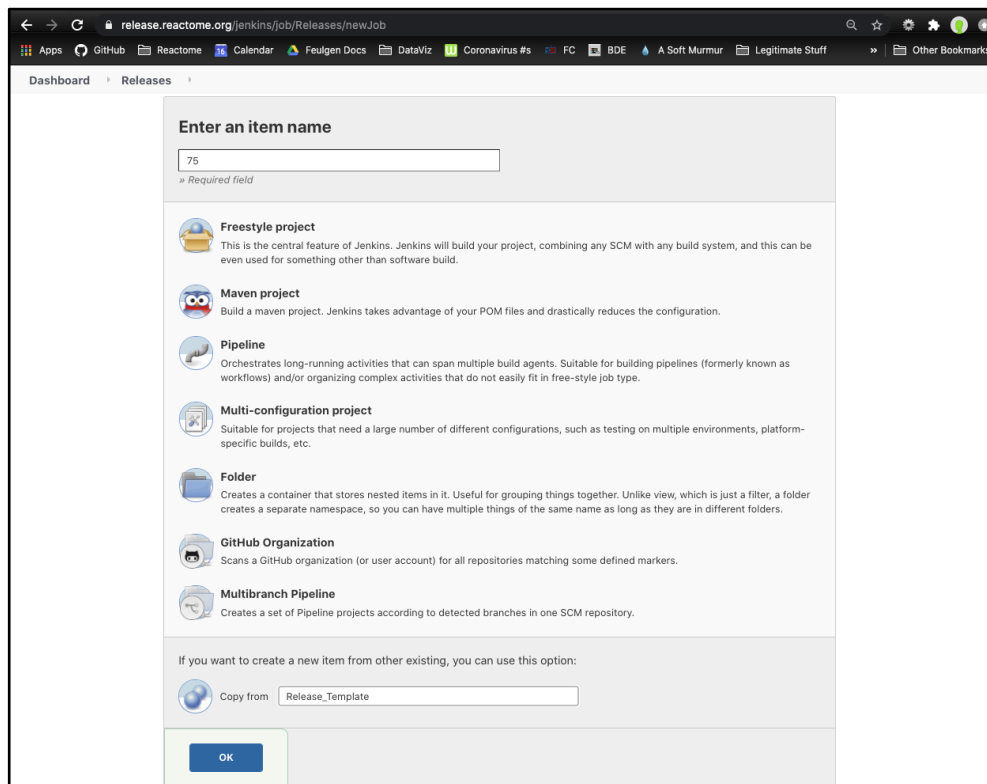
1) From the *Dashboard*, navigate to the *Releases* folder.



The screenshot shows the Jenkins web interface at the URL `release.reactome.org/jenkins/job/Releases/`. The left sidebar contains navigation links: Up, Status, Configure, New Item, Delete Folder, People, Build History, Project Relationship, Check File Fingerprint, Move, Rename, Credentials, and New View. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing 1 idle and 2 idle executors). The main content area is titled 'Releases' and features a table with columns: S, W, Name, Last Success, Last Failure, and Last Duration. The table lists four items with names 71, 72, 73, and 74, all showing 'N/A' for success and failure. Below the table, there is an 'Icon: S M L' section and a 'Legend' with three options: 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'. The bottom right corner of the page indicates 'REST API' and 'Jenkins 2.260'.

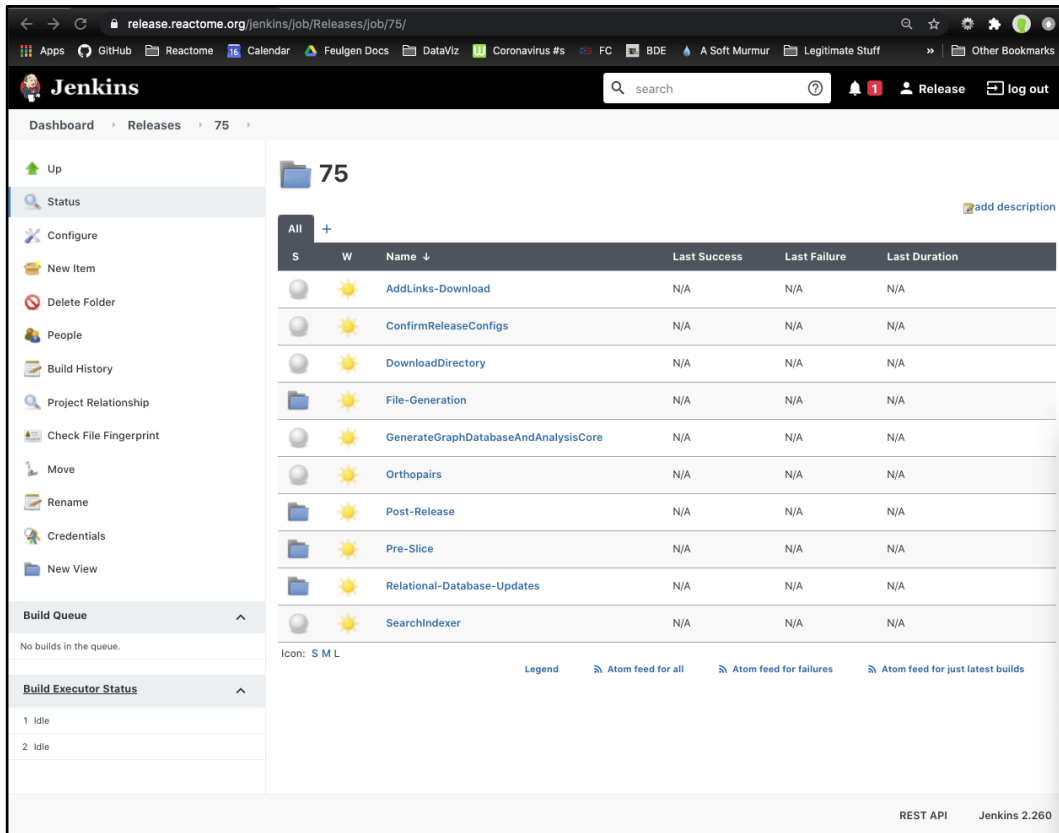
S	W	Name	Last Success	Last Failure	Last Duration
		71	N/A	N/A	N/A
		72	N/A	N/A	N/A
		73	N/A	N/A	N/A
		74	N/A	N/A	N/A

2) Select *New Item* from the left-hand side. This will bring you to the project creation page. Under *Enter an item name*, you will want to enter the release number (eg: '75'). You can ignore the list of project types, but at the bottom, there is an option to *Copy from*. In this field, you will want to enter 'Release_Template'. Click OK.



- 3) After clicking OK, you will be brought to a configuration page. Since we are copying from the release template, you don't need to worry about this. Click *Save* at the bottom of the page.

- 4) You should then arrive at a page with a number of 'jobs' with sunny icons beside them. Great! You've successfully created a new Release folder. The folder should have a path of *Dashboard > Releases > XX*, where **XX** is the release number you input just now when you created the project. **Here, and throughout this document, XX will be 75.**



Adding or Updating the Master Configuration File

So far we've updated Jenkins plugins and created a new Release folder. Now we will add the credentials/configuration file that will be used by Jenkins throughout the Release process.

At this point, database credentials (db.name) should point to **gk_central**.

- 1) Download the configuration file, found in the *release-jenkins-utils* GitHub repository: [link](#).
- 2) Change the cosmic release to the latest release based on this url:
https://cancer.sanger.ac.uk/cosmic/file_download/GRCh38/cosmic
- 3) This file will specify important credentials and other properties that are used by Releases' many steps. These properties will need to be filled out by you. If you are unsure of a certain property, ask a fellow developer familiar with the process for help in filling it out. The template shouldn't change often, so you may be able to get a copy of the file from a previous release. **Make sure you update the 'dateOfRelease', 'personId', and 'releaseNumber' properties!**
 - The 'dateOfRelease' can be found in the 'Release' column of the Release Schedule page at: <https://docs.google.com/spreadsheets/d/13rRNhz8kHlk-GIEaTYO86RAdbGyZs7H8iq41ldpiyo/edit#gid=0>

- 4) Once this configuration file has been filled out and saved, you can upload them to the credentials. **Make sure you are in the Release folder you created** (*Dashboard > Releases > XX*), select Credentials on the left-hand side.
- 5) You should see a list of credentials, including various login credentials. The values of these can't be viewed here. Rather, they are used by Jenkins to access the corresponding software. Look for the *Config* credential and click on its *Name* column (click where the red is highlighted in the below image).

The screenshot shows the Jenkins web interface at the URL `release.reactome.org/jenkins/job/Releases/job/75/credentials/`. The left sidebar contains navigation options like Up, Status, Configure, New Item, Delete Folder, People, Build History, Project Relationship, Check File Fingerprint, Move, Rename, Credentials, Folder, New View, Build Queue, and Build Executor Status. The main content area is titled 'Credentials' and displays a table of credentials for the 'Releases > 75' store. The table has columns for Type (T), Store (P), Domain, ID, and Name. The 'Config' credential is highlighted with a red box, and its name is expanded to show a redacted path: `reactome-jenkins-release-config-template.properties (This config file needs to be downloaded from release-jenkins-utils, filled out, and then uploaded here prior to running anything for release)`. Below the table, there are sections for 'Stores scoped to Releases > 75' and 'Stores from parent'.

T	P	Store ↓	Domain	ID	Name
🔑	🏠	Jenkins	(global)	mysqlUsernamePassword	piper/*****
🔑	🏠	Jenkins	(global)	jenkinsKey	jenkins/***** (For HTTP requests)
🔑	🏠	Jenkins	(global)	mysqlCuratorUsernamePassword	curator/*****
🔑	🏠	Jenkins	(global)	github credentials	cookersjs/*****
🔑	🏠	Jenkins	(global)	neo4jUsernamePassword	neo4j/***** (Credentials for neo4j user)
🔑	🏠	Jenkins	(global)	solrUsernamePassword	admin/***** (Credentials for Release Solr User)
🔑	🏠	Releases > 75	(global)	Config	reactome-jenkins-release-config-template.properties (This config file needs to be downloaded from release-jenkins-utils, filled out, and then uploaded here prior to running anything for release)

Icon: S M L

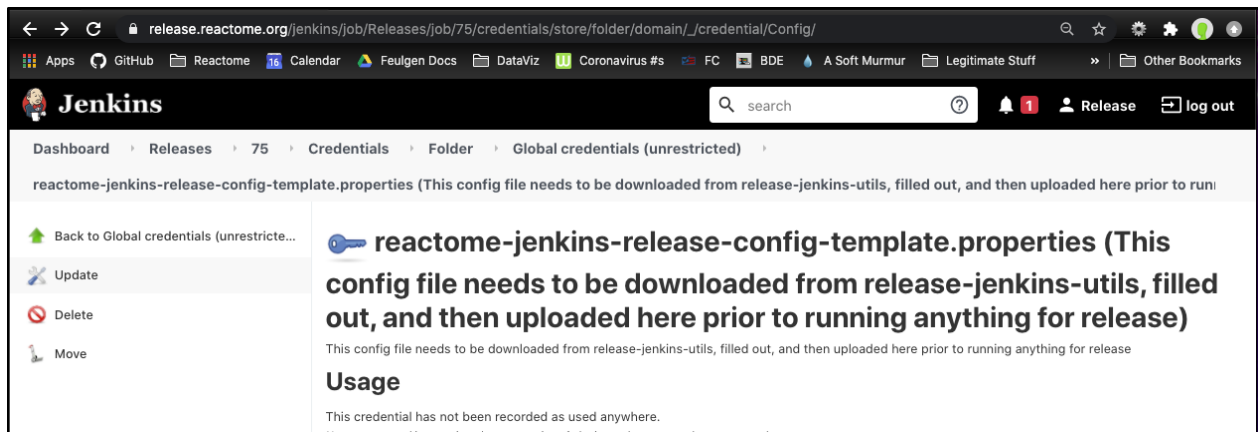
Stores scoped to Releases > 75

P	Store ↓	Domains
🏠	Releases > 75	🏠 (global)

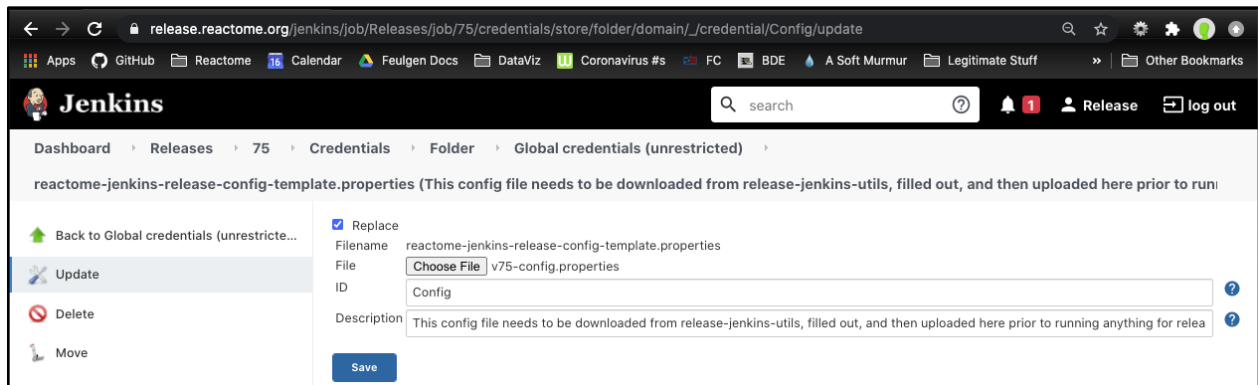
Stores from parent

P	Store ↓	Domains
🏠	Releases	🏠 (global)
🏠	Jenkins	🏠 (global)

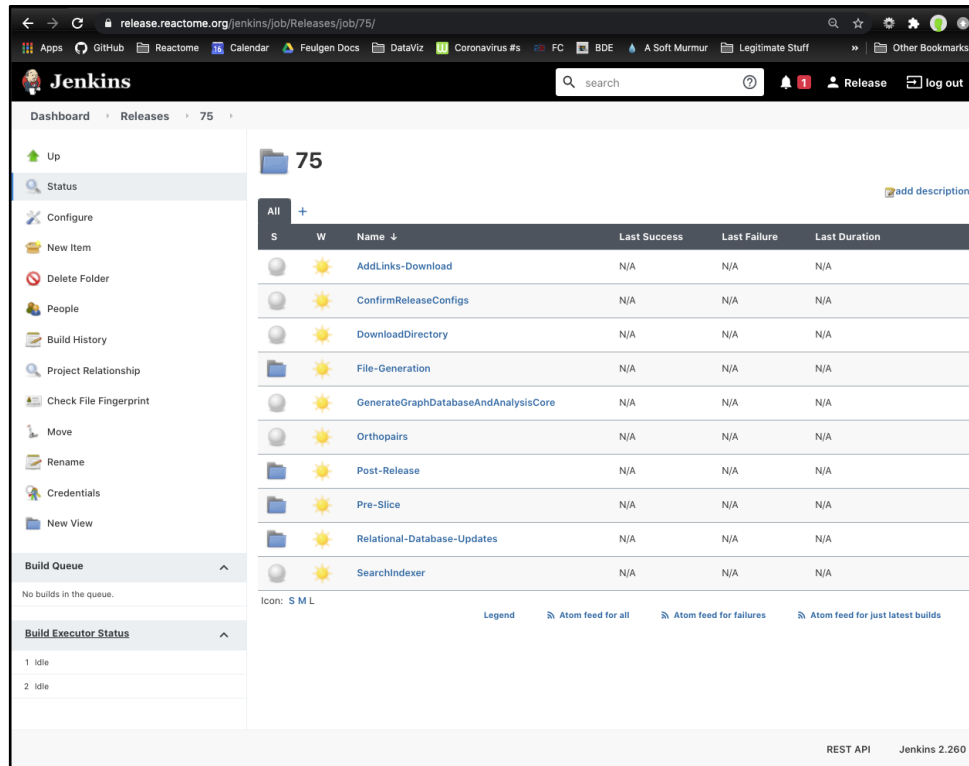
- 6) This will bring you to the configuration page for this credential. Click *Update* on the left-hand side.



- 7) Select the *Replace* option, and then *Choose File*. Find and select your filled-out configuration file from step 2. I named mine 'v75-config.properties' in this example. Click *Save* at the bottom once you've uploaded your file.



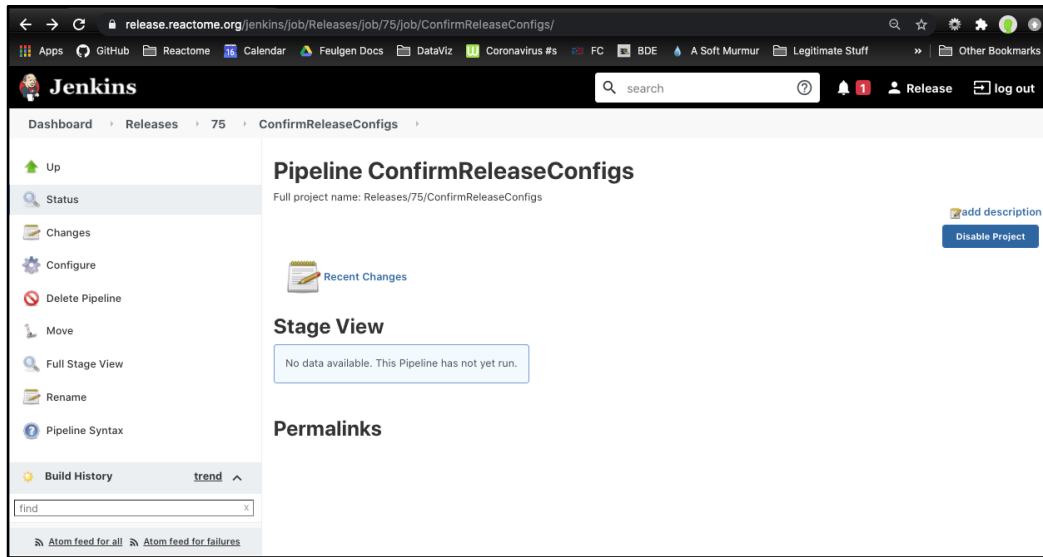
Great! Now we've created a new *Release* folder and updated the configuration file that will be used by the release steps. The final thing that needs to be done in the **Setup** phase is to make sure all of your release versioning is correct. Navigate back to *Dashboard > Releases > XX*.



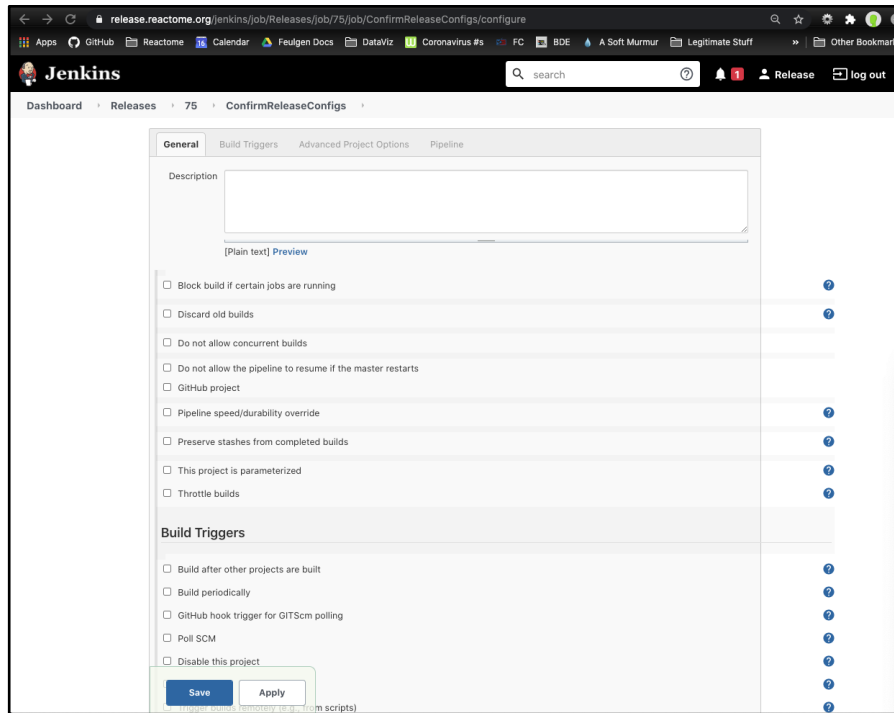
Confirm Release Configs

This is the first actual Jenkins *module*. This step checks that the *releaseNumber* variable in the uploaded configuration file matches the name of the Release folder you created. It also requests the user to input the release number, and checks that all these numbers match up. If they don't match up, it will fail and output where the discrepancy was.

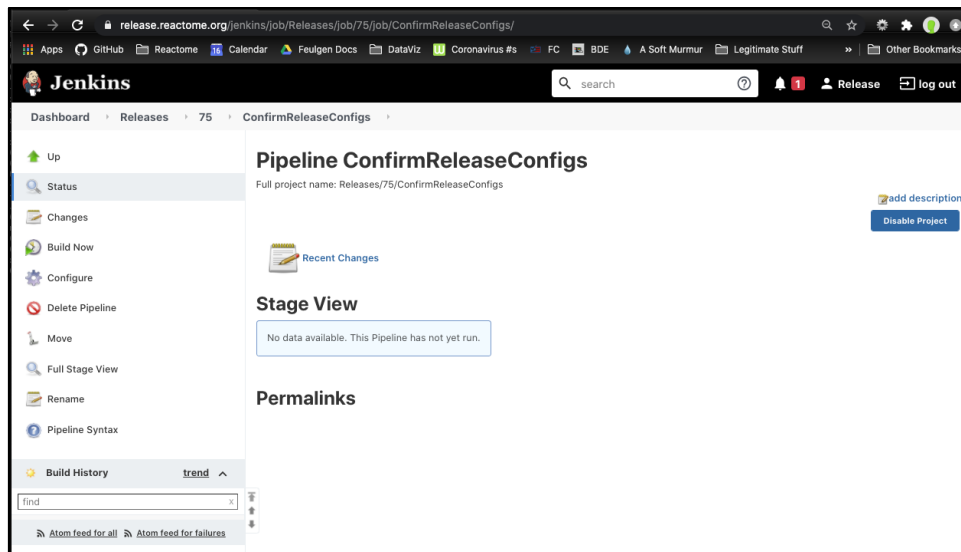
- 1) Select *ConfirmReleaseConfigs* from the jobs list at '*Dashboard > Releases > XX*', where *XX* corresponds to the Jenkins 'Release' folder you created during **Setup**.
- 2) This will bring you to the module page. Due to a quirk in Jenkins, there isn't any option to *Build Now* until after the default configurations have been saved. Click *Configure* on the left-hand side.



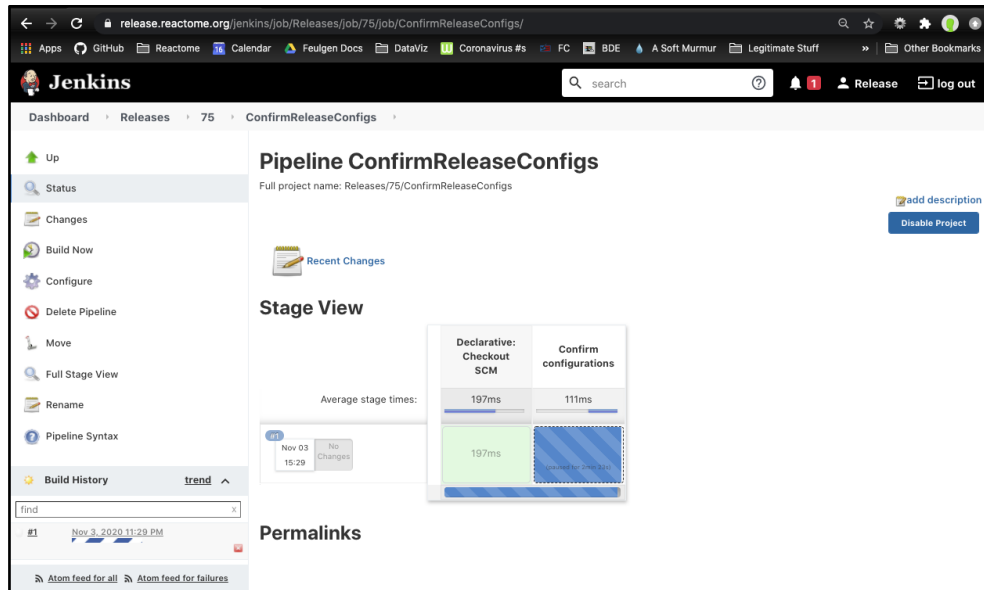
- 3) This page contains a lot of options that can be configured in a Jenkins job. At the time of writing, the only thing being utilized here is the section on the GitHub repository being used. Nothing needs to be updated, so just select Save at the bottom of the window.



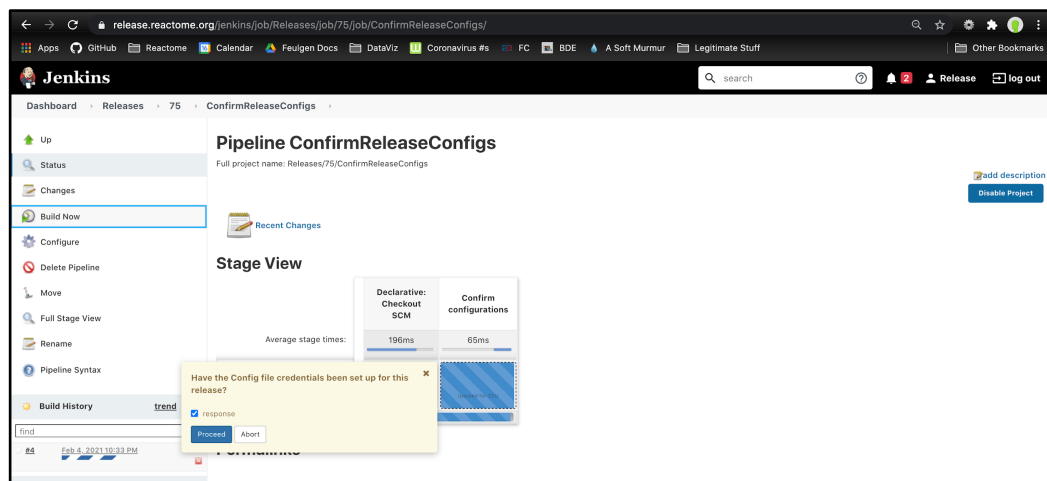
- 4) You should appear at the module page again. On the left-hand side, the *Build Now* option should now be visible. Select it.



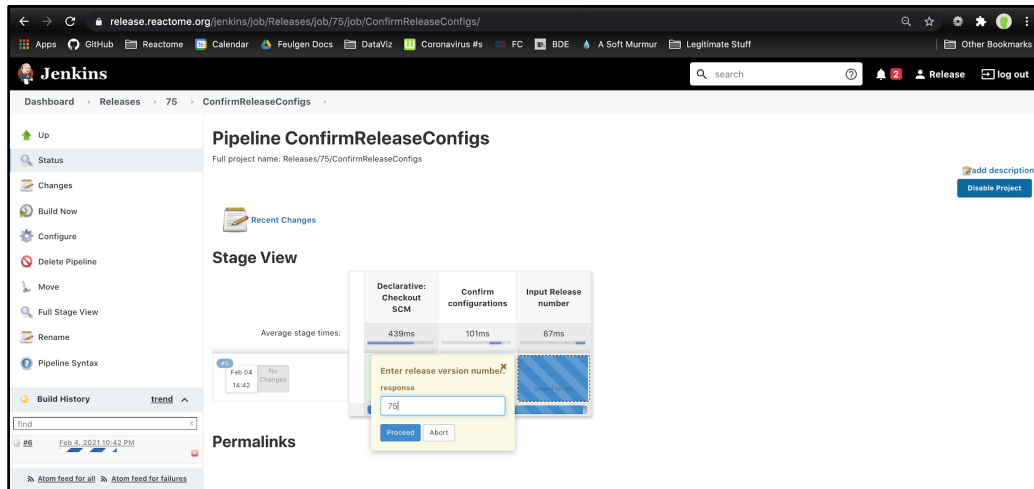
- 5) After a few seconds, you should see a progress bar appear at the bottom left (or a red circle if it failed). This step requires the user to answer a few questions before running, so it pauses the build until these have been answered.



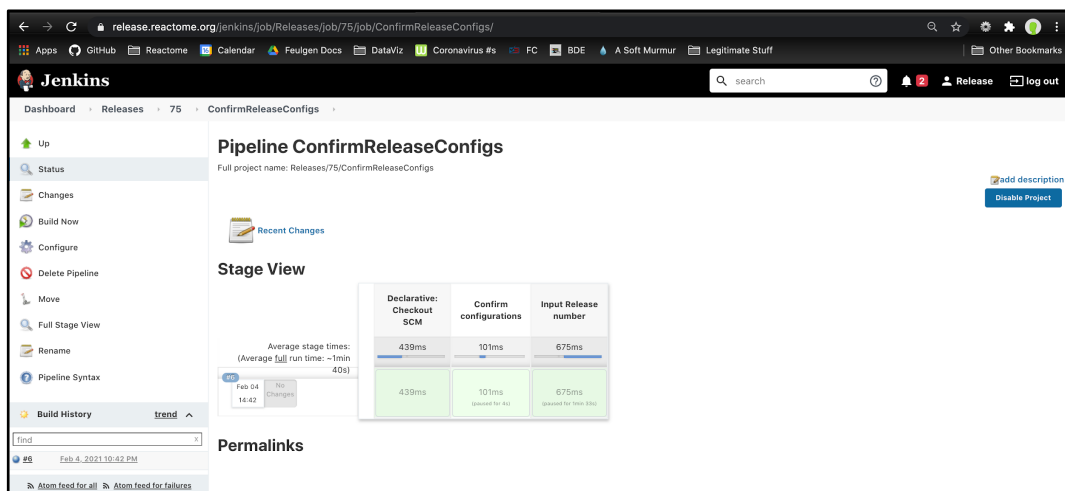
- 6) Double-click on the blue square underneath the *Confirm configurations* stage. A small window should appear asking if the config file credentials have been uploaded. Assuming you followed the **Setup** instructions above, answer with 'yes' and click *Proceed*.



- 7) The next stage will be a request for the user to input the release number. Fill this in in the pop-up window and then click *Proceed*.



- 8) If all configurations have been correctly set, then the step should finish. All boxes will be green if it was successful, or one will become red if it fails.



To review the console output of a build, you can hover your mouse beside the build number at the bottom left (in this case, **#1**). A little black arrow should appear when hovering near it. If you

click on this arrow, a menu will pop up for different build-specific pages you can navigate to. If interested, click *Console Output*.

Dashboard > Releases > 75 > ConfirmReleaseConfigs

Pipeline ConfirmReleaseConfigs

Full project name: Releases/75/ConfirmReleaseConfigs

add description
Disable Project

Recent Changes

Stage View

Stage	Declarative: Checkout SCM	Confirm configurations
Average stage times:	197ms	774ms
(Average full run time: ~21min 50s)	197ms	774ms (based for 21min 47s)

Permalinks

- Last build (#1), 22 min ago
- Last stable build (#1), 22 min ago
- Last successful build (#1), 22 min ago
- Last completed build (#1), 22 min ago

REST API Jenkins 2.260

Dashboard > Releases > 75 > ConfirmReleaseConfigs > #1

Console Output

Started by user Release

Obtained jenkins/Jenkinsfile-confirm-release-configs from git <https://github.com/reactome/release-jenkins-utils>

Running in Durability level: MAX_SURVIVABILITY

Loading library reactome-jenkins-utils#feature/shared-groovy-library

Examining reactome-jenkins-utils

Attempting to resolve feature/shared-groovy-library as a branch

Resolved feature/shared-groovy-library as branch feature/shared-groovy-library at revision 3f40ba1c776233e6317fcb4a914f2144058d600

The recommended git tool is: git

using credential github credentials

> git rev-parse --is-inside-work-tree # timeout=10

Fetching changes from the remote Git repository

> git config remote.origin.url <https://github.com/reactome/release-jenkins-utils.git> # timeout=10

Fetching without tags

Fetching upstream changes from <https://github.com/reactome/release-jenkins-utils.git>

> git --version # timeout=10

> git --version # 'git version 2.17.1'

using GIT_ASKPASS to set credentials

> git fetch --no-tags --progress -- <https://github.com/reactome/release-jenkins-utils.git> +refs/heads/feature/shared-groovy-library:refs/remotes/origin/feature/shared-groovy-library # timeout=10

Checking out Revision 3f40ba1c776233e6317fcb4a914f2144058d600 (feature/shared-groovy-library)

> git config core.sparsecheckout # timeout=10

> git checkout -f 3f40ba1c776233e6317fcb4a914f2144058d600 # timeout=10

Commit message: "Add emailing method, without attachments"

First time build. Skipping changelog.

[Pipeline] Start of Pipeline

[Pipeline] node

Running on Jenkins in /var/lib/jenkins/workspace/Releases/75/ConfirmReleaseConfigs

[Pipeline] {

[Pipeline] stage

Pre-Slice

Prior to the database updates below, steps 1-7 in the Editorial Release SOP [Preslice checklist](#) (*Appendix R2*) must be completed.

Wait for Managing editor (Lisa) to send out an email telling people not to make updates to gk_central. Once that email has been sent out we can start the pipelines.

Pre-Slice encompasses 4 steps currently: *UniProtUpdate*, *GOUpdate*, *ChEBIUpdate*, and *COSMICUpdate*. All of these steps take place in *Dashboard > Releases > XX > Pre-Slice*. They are much simpler to run than the **Setup** steps.

Uniprot Update (4hrs)

You will need to navigate to the steps page and select *Configure > Save* in order to bring up the *Build Now* option. Once it appears, you should be able to just click that and run the step.

Post-step validation: You should receive an email from Jenkins with a 'uniprot.wiki' file attached. This file will need to be uploaded to the DevWiki (following the email's instructions) and then validated by Curation. You can also look over logs and reports and compare them with the previous release if you want to review this step in more detail.

GO Update (20 mins)

You will need to navigate to the steps page and select *Configure > Save* in order to bring up the *Build Now* option. Once it appears, you should be able to just click that and run the step.

Post-step validation: You should receive an email from Jenkins with a 'go-update-v82-reports.tgz' file attached. The contents of this file will need to be uploaded to the Reactome GDrive and its URL updated in the DevWiki (following the email's instructions) and then they will need to be validated by Curation. You can also look over the logs and reports and compare them with the previous release if you want to review this step in more detail.

ChEBI Update (5 mins)

You will need to navigate to the steps page and select *Configure* > *Save* in order to bring up the *Build Now* option. Once it appears, you should be able to just click that and run the step.

Post-step validation: You should receive an email from Jenkins with a 'chebi-update-v75-reports.tgz' file attached. The contents of this file will need to be uploaded to the Reactome GDrive and its URL updated in the DevWiki (following the email's instructions) and then they will need to be validated by Curation. You can also look over the logs and reports and compare them with the previous release if you want to review this step in more detail.

COSMIC Update (45 mins)

Make sure you have at least 35GB of disk space free

Update reports can be found [here](#).

Creating Slice Database

Prior to creating a slice database, [preslice checklist](#) (Appendix 2) in the [editorial release SOP](#) must be completed.

Specific instructions for creating the slice database on curator and moving it to release

1. ***If taking the final slice**, see [Final Slice section below](#)*
2. If taking the second (or later):
 - a. On curator: `cd /home/weiserj/slicingTool/`
 - b. Go to step 3e below
3. If taking the first test slice:
 - a. On curator, `cd /home/weiserj/slicingTool/`
(Will make it so that we are working from `usr/local/reactomes/Reactome/production/Release/slicingTool/` in the future)
 - i. Create a file=`verXX_topics.txt` with current topics
 - ii. Update `slicingTool.prop`
 1. `releaseTopicsFileName=verXX_topics.txt`
 2. `slicingDbName=test_slice_<release-version>`
 3. Update `releaseNumber`
 4. Get release dates from:
<https://docs.google.com/spreadsheets/d/13rRNhz8kHlk-GIEaTYO86RAdBGyZs7H8iq41ldpiyo/edit?gid=0#gid=0>
 5. `runCharacterFixer=true`
 6. Verify other attributes are as follows:

- a. *needUpdateTrackers=True*
- b. *uploadUpdateTrackersToSource=False*
- c. *setReleasedInStableIdentifier=False*
- d. *updateReviewStatusToSource= False*

- b. If taking the **final slice** , follow the step [here](#).
 - c. Run "screen ./runProjectSlicingTool.sh"
 - d. Run "echo \$?" Immediately after the slicing command
 - i. If you get the response "0" it finished successfully
 - e. Check done.txt
 - f. Check SlicingTool.log for any errors (notably "Diagram X has a pathway node has no diagram associated")
 - g. Check SlicingTool.log for any errors (notably "Diagram X has a pathway node has no diagram associated")
 - h. Check database
 - i. Log into MySQL using the "[generic curator tool](#)" login and password.
 - ii. use test_slice_<release_version>;
 - iii. SELECT count(*) FROM DatabaseObject;
 - 1. Should be above 500,000
 - 2. SELECT * FROM _Release;
 - 3. This should provide the correct release and release date
 - iv. Exit mysql
4. Again...If taking the **final slice** , follow the step [here](#).

Move slice database over from curator to the release

1. If this is NOT the first test slice of the given release, first do these steps **in the /tmp directory** on [release](#)

```
rm test_slice_<release-version>.sql.gz
```
2. If this is NOT the first test slice of the given release, first do these steps **in the directory on [curator](#) where you create database dumps (r)**

```
rm test_slice_<release-version>.dump
rm test_slice_<release-version>.sql.gz
```
3. On [curator](#) Dump test slice: In **/home/weiserj/slicingTool/**,

```
mysqldump -ucurator -p test_slice_<release-version> > test_slice_<release-version>.sql
```
4. gzip .sql file

Run "gzip test_slice_<release-version.sql"

5. Scp the file onto the machine (first to your local machine and then to release)
 - a. On local machine (in directory where you will be sending the database):
scp -i ~/.ssh/matthews.pem
matthews@curator.reactome.org:/home/weiserj/slicingTool/test_slice_XX.sql.gz
/Users/xxxxx/directoryX/test_slice_xx.sql.gz
 - b. On local machine (in directory with the zipped database):
scp -i ~/.ssh/matthews.pem test_slice_92.sql.gz
matthews@release.reactome.org:/tmp/test_slice_92.sql.gz

Install test slice on release

Ssh onto release to directory database was put: /tmp

Go into MySQL and remove the database and create a new database

Run "mysql"

Inside MySQL run:

- i. DROP database slice_test;
- ii. CREATE database slice_test;
- iii. show databases;
- iv. Exit

Slice QA

Prior to release, a series of quality control check are run on the curated content in the slice database. These check for: data internal consistency, required experimental evidence, appropriate manual review of inferred/new/revise data, valid external links, consistency between pathway diagrams and database content, readability and navigability of pathway diagrams, and proper contributor attribution. A complete list of the quality checks are listed in [here](#) in *Appendix R6*.

After the pre-slice steps have been completed, Editorial release manager may take a slice and may want to view the diagram and EHL changes. This can be done by running slice qa pipeline in Jenkins. It may also need to be run multiple times if Lisa needs to make multiple test slices.

Checks prior to Jenkins pipeline:

- Event where stableIdentifier "released" is null and releaseDate != current release date (~~this step will be added to release QA pipeline check~~).
- Run `unreleased_instances_with_released_stable_id.pl` script (in `/usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update`) to look for released events that have gone missing (set to `do_release false` or deleted from database) since last release using (~~to be added to release QA pipeline and weekly QA~~)
- Look for events that are inferred but whose "inferredFrom" event is missing from slice (~~this step will be added to release QA pipeline and weekly QA~~).
- Run `compartment_info.pl` in the `/usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update` directory. Identifies new compartments in use in Reactome (Send to Peter for review and add/verify surroundedBy attribute assignment. When this has been done, **send email to dev with title "Developer Action Item VerXX: new compartments"** In VXX, there is a new compartment. This will need to be taken into account by the reaction diagram drawing feature on the contents details page.
- Run `deleted object in diagram check` on `gk_central` to look for entities/events deleted from event hierarchy but still present in diagrams (is now part of release QA pipeline checks, but missing objects cause weekly QA to fail).
- Update species list if any new species have been added/removed from orthoinference.

Steps for running SliceQA in Jenkins (on every slice)

Jenkins SliceQA pipeline

Accessing the pipeline

https://release.reactome.org/jenkins/job/SliceQA/job/Run_Slice_QA/

Specify just the number of the version in build parameters (i.e 93, not V93)

Running the pipeline

The first run must be completed before the new/updated Illustrations files (EHLD/Static) can be uploaded to the release server by the Illustrator

If first run, **update qa.properties cutoff date** ([final slice date for previous release](#)) here:
/home/awright/gitroot/release-qa

1. Verify slice file in /tmp with file named test_slice_<version>.sql.gz (e.g. test_slice_84.sql.gz)
 - a. cd /tmp
 - b. ls -l
2. In Jenkins interface click “Build with parameters” (use downward arrow to the right of Run_Slice_QA)
 - a. It will ask for the release version. Put the release version # (Use the # only. E.g 92 not V92)
3. Wait for it to complete
 - a. Look through output in Jenkins for each step to make sure ran correctly
 - b. Copy the report log files from the locations below to local computer to analyze further
 - i. Release-qa
 1. /var/lib/jenkins/workspace/SliceQA/Run_Slice_QA/release-qa/output
 - ii. Command-line-runner
 1. /var/lib/jenkins/workspace/SliceQA/Run_Slice_QA/command-line-runner/QA_Output/
 2. Graph-qa files
 3. /var/lib/jenkins/workspace/SliceQA/Run_Slice_QA/graph-qa/reports
 - iii. Diagram converter
 1. /var/lib/jenkins/workspace/SliceQA/Run_Slice_QA/diagram-converter/reports

Second run of the pipeline

1. **Email the illustrator to upload files Illustration files to release. Only needs to be done once, after the first run of pipeline** The protocol for file transfer can be found here (link to be added by Cris).
2. After the second run, check [new/revised Illustrations](#) (see appropriate release tab) on release.reactome.org.

Final Slice steps

Prepare clean slice (last test slice before final slice):

Curator tool reviewStatus check (This should be done only for the clean slice just prior to final slice)

- ~~Run the curator tool review status QA check~~ on the slice database ~~on curator (db=test_slice_XX).~~
- ~~Dump list in curator tool and create googledoc~~
- (Skip this section :See note in margin for current approach) Assign to curators ** Curator must check out instances AND fully extract all instanceEdit instances in the project as well as all ReviewStatus instances. Note: If the only thing that changed in a flagged pathway is that unreleased reaction(s)/event(s) were added, then nothing needs to be done. The slicing script checks if the hasEvent contents are unchanged relative to the gk_current (live site) and if they are the same, it keeps the greater of the two reviewStatuses. If you are unsure, you can check by looking at the flagged pathway in test_slice_XX.
 - ~~Be sure that the reviewed/internally reviewed attributes are added in chronological order (most recent at the bottom of the list of attributes).~~

Curator tool NewStableIdentifier check

- ~~Run the curator tool NewStableIdentifier check in the curator tool on gk_central. Looks for wrong stableID prefixes for new instances~~

Just prior to final slice

- ~~Verify in last test slice no events with <2 stars~~
- ~~Verify there are no literatureReferences in the slice that have retraction_status that contains Retracted or "no_use".~~
- ~~Take a dump of gk_central~~
 - ~~cd /home/weiserj/slicingTool/~~
 - ~~mysqldump ucuration p gk_central > gk_central_20240306b.sql~~

Run final slice on curator

Update slicingTool.prop for final slice on curator

- ~~Go to slicingTool directory: cd /home/weiserj/slicingTool/~~
- ~~Update slicingTool.prop parameters:~~

- ~~needUpdateTrackers=true~~
Controls if final slice gets update tracker
This one can be used for testing as it only writes to slice
 - ~~uploadUpdateTrackersToSource=true~~
Controls if it writes back to gk_central or not
 - ~~setReleasedInStableIdentifier=true~~
 - ~~updateReviewStatusToSource=true~~
- Run “screen ./runProjectSlicingTool.sh”
 - Check slice as above for previous test slices ([steps f-i](#))
 - Check that _updateTracker changes have registered.

Reset slicingTool.prop parameters immediately after final slice QA:

- ~~Check updatetracker instances and stableId release attributes in gk_central~~
- Update slicingTool.prop (out of final slice mode)
 - ~~needUpdateTrackers=true~~
 - ~~uploadUpdateTrackersToSource=false~~
 - ~~setReleasedInStableIdentifier=false~~
 - ~~updateReviewStatusToSource=false~~

Move slice database over from curator to the release

1. Remove previous test_slice file from the **/tmp directory** on **release**
 - a. rm test_slice_<release-version>.sql.gz
2. Remove the previous test_slice file **in /home/weiserj/slicingTool/** on **curator**
 - a. rm test_slice_<release-version>.dump
 - b. rm test_slice_<release-version>.sql.gz
3. On **curator** Dump test slice: In **/home/weiserj/slicingTool/**,
 - a. mysqldump -ucurator -p test_slice_<release-version> > test_slice_<release-version>.sql
4. gzip .sql file
 - a. Run “gzip test_slice_<release-version>.sql”
5. Scp the file onto the machine (first to your local machine and then to release)
 - a. scp -i ~/.ssh/matthews.pem test_slice_<release-version>.sql.gz
matthews@release.reactome.org:/home/matthews/test_slice_<release-version>.sql.gz

Install test slice on release

1. ssh to release
2. cd /tmp
3. Go into MySQL and remove the database and create a new database
 - a. Run "mysql"
 - b. DROP database slice_test;
 - c. CREATE database slice_test;
 - d. show databases;
 - e. Exit

Run Jenkins SliceQA pipeline with final slice

- ~~As described [above](#).~~

Populate slice_test database with release final slice

This step should be done after QA on final slice is complete.

In [/home/weiserj/slicingTool](#) on curator:

Run bash ./update_test_slice_from_release.sh \$DB \$USER \$PASS

- The parameters are from slicingTool.prop
 - DB = slicingDbName (e.g. test_slice_84)
 - USER = slicingDbUser (curator)
 - PASS = slicingDbPwd
- Check rotation worked
 - Login to mysql (curator credentials)
 - mysql> use test_slice_from_release
 - mysql> select max(releaseNumber) from _Release; (should show current database)

Notify Developers to start "Update release_current: Relational Database" step

- **~~DO NOT reopen gk_central until updateStableID and UpdateDOI steps have been completed!~~**
- **~~Check stableId and _updateTrackers are updated~~**

Reopen gk_central

- ~~Email internal reopening gk_central~~

QA cleanup

- * If this is the 3rd quarter (September) release, Ask Guanming to run the retraction check.
- Following the final slice, Managing editor updates necessary resource files for the next release cycle QA (weekly and release) by following steps manually by editorial release manager

These are the files that need to be updated on curator:

- /usr/local/reactomes/Reactome/production/qa/weekly/CuratorQA/resources/qa.properties
- /usr/local/reactomes/Reactome/production/qa/weekly/ReleaseQA/resources/qa.properties
- /usr/local/reactomes/Reactome/production/qa/weekly/SlicingTool/slicingTool.prop (update release version, releaseDate, and previous releaseDate)
- /usr/local/reactomes/Reactome/production/qa/weekly/SlicingTool/topics.txt (if necessary)

Steps:

- Create skip lists for QA tool based on previous release and update these in the .../CuratorQA/resources/ and .../ReleaseQA/resources/ directories
- Run scripts to identify proteins in gk_central that have not been released
 - On reactomerelease, under /usr/local/gkbdev/scripts/release/website_files_update there now a script called 'uncurated_proteins.pl' you can run to get a list of uncurated protein in gk_central by default by typing "perl uncurated_proteins.pl. You can also type '-h' or '-help' to see options for running against other databases. The file lists uncurated proteins (including isoforms) but there is a unique list (parent form) at the bottom of the file. [br] [br]

Check for proteins in gk_central that have not been released can be found [here](#).

- Run script to identify events and entities that were previously released but now missing from live site: The script is on reactomerelease under the /usr/local/gkbdev/scripts/release/website_files_update and can be run by typing 'perl unreleased_instances_with_stable_id.pl'. You can add '-h or -help' to see how to change the default options if you need to do that. By default, the script assumes the curation database is gk_central and the release database is the most current test_reactome_XX.
- unreleased_EWAS.pl The script looks for EWASs that have been created but never released. It is on reactomerelease under the /usr/local/gkbdev/scripts/release/website_files_update and can be run by typing 'perl unreleased_EWAS.pl'

Editorial Release document preparation

After the final slice, the managing editor prepares the website files and other documents needed for release as described [here](#) in the Release documentation section of the Editorial Release SOP in *Appendix R1*.

Update release_current: Relational Database

This phase encompasses all updates that will be performed on the MySQL 'release_current' database. It includes 6 steps, run in roughly the following order: *UpdateStableIdentifiers*, *UpdateDOIs*, *Orthoinference*, *AddLinks-Insertion*, *OrthoinferenceStableIdentifierHistory*, and *BioModels*. The order isn't rigid. *UpdateDOIs* can be run at any point after *UpdateStableIdentifiers* (though the earlier the better as 'gk_central' will be closed until both *UpdateStableIdentifiers* and *UpdateDOIs* is run), and *OrthoinferenceStableIdentifierHistory* can just be run before or after *AddLinks-Insertion*.

Before starting Relational-Database-Updates, take a server snapshot.

- 1) Log into the AWS Management Console
- 2) Go to EC2
 - a) Select "snapshots" on left hand side
 - b) Click on create snapshot
 - i) Select instances
 - ii) Select "Release pre <release-version>" server (i-01a7c80f57d5525cd, at time of writing)
 - iii) Click "Submit"
 - c) Find it in list of snapshots and change name for snapshot to "Release"
- 3) This will likely take around 1-2 hours.
- 4) Once the 'Status' has changed from 'Pending' to 'Completed', you can proceed with Release.

Update Stable Identifiers (50 mins)

This step checks if any StableIdentifiers have changed between releases. Any that have, as judged by the presence of new instance edits in the 'modified' attribute of these instances, will have their 'identifier' attributes incremented.

Navigate to the UpdateStableIdentifiers step page in Jenkins at '*Dashboard > Releases > XX > Relational-Database-Updates*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Post-step validation: This step is a little complicated to validate, as it involves using the CuratorTool (or MySQL terminal if you are knowledgeable enough - I am not). Below are instructions on how to do that (will be moved to a separate tutorial in future).

- Check output for number of identifiers updated
- Check logs for output from StableIdentifierVersionMismatch
- Check slice_final directory for new DBs

Connecting to MySQL on a different server using ssh

- 1) Set up an ssh port connection between your computer and the release server. It requires you to have access to a key/pem file for sshing. Use the following command, substituting your pem file and username:

```
ssh -i jcook.pem -N -L 1234:localhost:3306 jcook@release.reactome.org
```

- 2) Open CuratorTool (can be downloaded [here](#)).
 - 3) At the top, Select '*Database > Database Browser > Schema View*'.
 - 4) A window should appear with login credentials. Enter the following:
 - o Database Host: localhost
 - o Database Name: release_current or release_previous
 - o Database Port: 1234 (or whatever port you chose after the -L tag above)
 - o User Name: Release server MySQL username (eg: piper).
 - o User Password: Release server MySQL password.
- In Jenkins, navigate to the console logs of the successful *UpdateStableIdentifiers* run or download them from s3://reactome/private/releases/74/update_stable_ids/logs/. If you are looking in the console, chances are the logs are truncated, and so you will need to select 'Full Log' from the top of the window. If you find they load slowly or the browser is having a tough time, I recommend just downloading them from S3. Once you can see the logs, do the following:
 - 1) Search for the phrase 'Incrementing'. This should have something that looks like 'Incrementing [StableIdentifier:5694847] R-HSA-5619084.4'.
 - 2) Now, take the DBID (5694847 here) and look for it in the CuratorTool, which should be set up as described in the above '**Connecting to MySQL on a different server using ssh**' section. I recommend just pasting the DBID in the blank square under 'Search Instances':

Search Instances

Choose Class: DatabaseObject

Choose Attribute: DB_ID

Attribute Value: Equals 5694847

Search Search More...

- 3) Click 'Search' and you should see an instance appear in the 'Found instances' column in the middle of the program. Select it. The right-most 'Instance Properties' column should show instance information.
 - 4) Depending on which database you are looking at (release_current or release_previous), you will be looking for different things:
 - **release_current:** The StableIdentifier 'identifierVersion' should be +1 from what the logs showed. If you were looking at 5694847 from the example, it should show 5.
 - **release_previous:** The StableIdentifier 'identifierVersion' should be the same as what the logs showed.
 - 5) Repeat this for a few more instances by searching in the logs for the 'Incrementing' string. I recommend finding cases where the identifierVersion was 1, and cases where the identifierVersion was well past 1. You will need to either bring up multiple CuratorTool windows, to look at both 'release_current' and 'release_previous', or just look at all instances in one database first for expected behaviour, before opening the CuratorTool to the other database.
- Another good sanity check is to go to the end of the logs and look for '### Stable Identifiers were updated'. Usually this ### number should be in the hundreds or maybe thousands (if Curation was particularly productive). Basically, as long as it's not a strangely low number (say less than 100), you should be good to proceed. If it is low, I recommend reaching out to Curation to find out why. Issues with this step or UpdatedDOIs are best resolved with Curation.

In output from "java -Xmx12000m -jar target/update-stable-ids-*-jar-with-dependencies.jar \${ConfigFile}"

Look for incrementing

Put in StableIdentifier and look for incremented ID

mysql -upiper -p

USE slice_current;

select * FROM StableIdentifier WHERE db_id = 369339

Update DOIs (7 mins)

This step updates both *gk_central* (on the curator server) and *release_current* databases. It first looks for Pathway instances. More specifically, it looks at the 'doi' attribute in each Pathway, and finds any that don't start with the string '10.3180' (the DOI prefix) in each database. This indicates that they need to be updated with a new DOI. Generally, they will be filled with the 'needs DOI' string.

Pre-step: Before running the step, it's good to confirm for yourself which DOIs will be updated. Access the MySQL terminal on **both** *release.reactome.org* and *curator.reactome.org*, and run the following commands:

```
Use database;
SELECT * from Pathway where doi NOT REGEXP '^10.3180' order by db_id;
```

For *release.reactome.org*, 'database' will be *release_current*.

For *curator.reactome.org*, 'database' will be *gk_central*.

Below is an example that checks *gk_central* (from v75). See how the 'doi' rows all contain the value 'needs DOI':

```
mysql> use gk_central;
[Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * from Pathway where doi NOT REGEXP '^10.3180';
```

DB_ID	doi	isCanonical	normalPathway	normalPathway_class	hasEHL
5693565	needs DOI	NULL	NULL	NULL	NULL
9006821	needs DOI	NULL	NULL	NULL	NULL
9607240	needs DOI	NULL	NULL	NULL	NULL
9659379	needs DOI	NULL	NULL	NULL	NULL
9663199	needs DOI	NULL	5693606	Pathway	NULL
9682385	needs DOI	NULL	9607240	Pathway	TRUE
9690406	needs DOI	NULL	NULL	NULL	NULL
9699150	needs DOI	NULL	5693606	Pathway	NULL
9706574	needs DOI	NULL	NULL	NULL	NULL

```
9 rows in set (0.00 sec)
```

The *release_current* database may have less hits than *gk_central*, as sometimes these Pathways haven't been released yet. If *release_current* has more hits though, that needs to be investigated by a curator. Check if the same thing happens in the *slice_test* and *slice_current* databases. If so, this will need to be investigated by Curation.

This is just to give you an idea of what Pathway instances will have their 'doi' attribute updated by the UpdateDOIs step.

Running UpdateDOIs

Navigate to the UpdateDOIs step page in Jenkins at '*Dashboard > Releases > XX > Relational-Database-Updates*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

UpdateDOIs first completes a 'test run' to determine which Pathway DOIs will be updated. It will then email this list, which is stored in a file called 'doisToBeUpdated-vXX.txt', where XX is the release number, which will need to be confirmed by Curation. Lisa will email a confirmation to Adam and Joel that she has reviewed the contents of the files.

Once Curation has confirmed the list of DOIs, we need to let Jenkins know it can proceed. Navigate to the *UpdateDOIs* page and double click the blue box under the 'User Input Required: Confirm DOIs' stage. This should cause a small window to pop up that prompts you to proceed once you've received Curation approval. Click proceed. If 'User Input' is unfamiliar in Jenkins, I recommend looking over the 'ConfirmReleaseConfigs' portion, which has pictures (This will be put into a more generic tutorial at a later date).

Runtime: About 5 minutes (excluding wait time for user input)

Validation

- 1) Re-run the MySQL query that you ran during the pre-step check on both the *gk_central* and *release_current* databases.
- 2) The query should return an empty set in *release_current*.
- 3) In *gk_central*, generally it should be an empty set but sometimes there are still some rows returned, if there are unreleased Pathway instances. If the *gk_central* query does return any hits, be sure to check that none of them are found in the 'doisToBeUpdated-vXX.txt' file that was emailed after the test run.
- 4) If *gk_central* returns any hits from 'doisToBeUpdated-vXX.txt', or if *release_current* returns any hits, then the logs will need to be looked at to determine what happened.

```
mysql> use release_current;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * from Pathway where doi NOT REGEXP '^10.3180';
Empty set (0.00 sec)
```

You should have received a second email confirming that UpdateDOIs *and* UpdateStableIdentifiers have completed. Once you've confirmed that UpdateDOIs has run properly, **let Curation know so they can reopen *gk_central*.**

Orthopairs (30 mins)

This step downloads protein/gene orthology data files from PANTHER and protein mapping files from other model organism databases (MGI, RGD, Xenbase and ZFIN). These mapping files are used to increase the number of positive hits in the orthology files, as PANTHER uses both Ensembl identifiers as well as those from model organism resources. There is also a statically stored identifier mapping file from <https://www.yeastgenome.org/>. We've been told that this file doesn't change very much, so we'd be OK to keep a static file then build a semi-complicated file downloader.

Navigate to the Orthopairs step page in Jenkins. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation: This step can be verified by comparing line counts of the generated Orthopairs files between releases.

- 1) Navigate to the console output of the step "Orthopairs file line counts". This will show a file-by-file line count comparison between releases, including the difference between the files.
- 2) Confirm that the total number of files are the same. This can be seen *above* the 'line count differences' phrase.
- 3) Scroll through each of the files and look at the 'Difference' in line counts between them, looking for those with a large difference (over 10%). If any are found, this will need to be inspected further by looking at the orthology files or logs of the program.
- 4) If the line counts look OK, we just want to confirm that the orthopairs files are visible in the S3 bucket. Log in to S3 and navigate to `reactome/private/releases/XX/orthopairs/data/orthopairs`. Confirm there is an 'orthopairs' folder that contains the same number of files as found in step 2.

Note: Be more explicit with confirmation instructions

Orthoinference (15-20hrs)

Orthoinference requires that the Orthopairs step has been successfully created for it to run.

Orthoinference takes the orthology mapping files that were created during Orthopairs and then creates an 'inferred' Pathway hierarchy from them for a variety of different model organisms (mouse, fly, zebrafish, etc.). It does this by taking all Human Reactions in the database,

breaking them down into their Proteins subunits. These Human Proteins are mapped to a UniProt identifier in the database, which may also be mapped to an 'ortholog' in the Orthopairs files of the species that is being inferred. If an ortholog does exist, then Orthoinference will create an 'orthologous' Protein instance in the database. If enough orthologous Proteins exist, then an orthologous Reaction is created. At the end of it all, a number of 'inferred' Reactions will exist, and a Pathway hierarchy is created using the Human Pathway hierarchy as a model. Such is Orthoinference!

In Jenkins, once Orthoinference has completed, a graph database will be created so that graph-qa can be run. This is critical to try and weed out any systemic issues that might exist in Reactome, as Orthoinference increases the number of database instances by more than 10x!

Navigate to the Orthoinference step page in Jenkins at '*Dashboard > Releases > XX > Relational-Database-Updates*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

For Orthoinference, there are two ways to quickly verify that it ran correctly.

- 1) Review the logs from the 'Orthoinference file line counts' step. This should show a comparison of the 'inferred' and 'eligible' file line counts, and compare them with those same files from the previous Release. Generally, you should see a small increase in the number of lines for the 'eligible' files, which reflects that new Reactions have been added to the database. This should be the same for the 'inferred' counts, but it also depends on the Orthopair files line counts. If you notice that the 'inferred' counts are reduced between releases, it is recommended that you see if the corresponding Orthopair files saw a decrease from this release.
- 2) At the end of Orthoinference in Jenkins, an email should have been sent that contains post-Orthoinference graph-qa results from this and the previous release. Confirm that there aren't any 'BLOCKER'-level issues and that there aren't many new 'HIGH', 'MEDIUM' or 'LOW'-level issues. At time of writing, there were generally a number of issues reported in those categories, but we have learned to live with them. If they look similar to the ones found in the previous release, chances are they OK. In the event that there are new issues that you are unsure of, it is best to consult with Curation before diving into the source of the reported issues. After reviewing the graph-qa files, it is good to get confirmation from Curation that the graph-qa results look acceptable.

A final item is to check the orthoinference report file. Orthoinference generates a file called 'report_ortho_inference_test_reactome_XX.txt', which contains a statement on how many reactions were inferred out of the eligible amount for each species. This file has existed since the Perl release days and is used to generate the [inferred events](#) page on reactome.org.

When Orthoinference was rewritten in Java, it was found that it performed better by inferring the 'heavier' (ie. more inferred Reactions; eg. mouse) species before the 'lighter' (eg. yeast) ones.

Despite this, Curation still required the file contents to be ordered the original way, so a small bash script just creates a re-ordered version of the file that matches the historical ordering.

Now that we know *why* that file exists, how do we check it? On the release server, navigate to **/usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update/**. Confirm that the following two files exist, and were made around the same time as Orthoinference completed:

- 1) report_ortho_inference_test_reactome_XX_sorted.txt
- 2) report_ortho_inference.txt

Run `diff report_ortho_inference_test_reactome_XX_sorted.txt report_ortho_inference.txt`. They should be the same. If they are different it means that the orthoinference steps did not complete. For extra confirmation, you can look at the contents of the report file, and confirm the numbers correspond to the line counts of the inferred/eligible files that were checked earlier.

Orthoinference Stable Identifier History (2hrs)

This stage doesn't technically need to follow Addlinks-Insertion, as it only requires that the Orthoinference step has finished successfully. (In future, consider rolling this into a massive 'orthoinference' step?)

OrthoinferenceStableIdentifierHistory should be much simpler than the Orthoinference or AddLinks processes, as it really only updates StableIdentifier instances in the *release_current* database, and updates the StableIdentifier history in the *stable_identifiers* database. At time of writing, it runs two Perl scripts, 'save_stable_id_history.pl' and 'old_stable_id_mapping.pl', as well as the usual StableIdentifier QA.

Navigate to the OrthoinferenceStableIdentifierHistory step page in Jenkins at '*Dashboard > Releases > XX > Relational-Database-Updates*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

After the step completes, navigate to s3 and download the log files from `s3://reactome/private/releases/XX/ortho_stable_id_history/logs`. There should be three types of logs:

- 1) 'old_stable_id_mapping' -> The log file itself doesn't need to be examined much, as long as it is a non-zero value. At time of writing, an *err* file is usually made that contains plenty of 'WARN' statements. Those are normal, but you should look for the string 'FATAL' to see if any serious errors were thrown.

- 2) 'Save_stable_id_history' -> Again, the log file does not usually need to be examined much, so long as it is a non-zero value, and there isn't any *err* file, you should be OK. If there is an *err* file, then it will need to be examined, as that is not usually the case (unlike *old_stable_id_mapping*).
- 3) 'OrthoStableIdentifierHistory' -> Check the *err* file for anything. It will be produced but most often it is a 0-byte file (after being gunzipped). The main log will normally show a couple messages ~15 minutes apart. As long as there were not any error messages thrown, you should be able to proceed.

Once you've checked all 3 log files and determined that they are OK, you can move on to the next step.

Generate Graph Database (3 hrs)

This step produces another graph database and analysis core from a version of the *release_current* database that includes the BioModels cross-references. It is essentially the same as the BioModels step, save for the creation of the 'models2pathways.tsv' file and the addition of cross-references.

Navigate to the GenerateGraphDatabase step page in Jenkins at '*Dashboard > Releases > XX*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

This step is also fairly easy to validate, requiring that you just confirm the new analysis bin file is being used, and that the PathwayBrowser for *release.reactome.org* is up to date.

In a browser, navigate to <https://release.reactome.org/PathwayBrowser/> and confirm that the release number (red box) is correct:

Stage View

Check
ConfirmReleaseConfigs
build succeeded

User Input:
Get FlyBase
date in file
URL and
update it in
basic-file-
retrievers.xml

Average stage times: 220ms 47ms

Please submit the YYYY_MM value in the filename of the file called 'fbgn_NAseq_Uniprot_fb_YYYY_MM.tsv.gz'. It can be found at 'ftp://ftp.flybase.net/releases/current/precomputed_files/genes'.

response

It should be in the format 'YYYY_MM' (eg. '2020_05')

Proceed Abort

Navigate to that URL and find the 'YYYY_MM' value from the 'fbgn_NAseq_Uniprot_db_YYYY_MM.tsv.gz' file:

← → ↻ Not Secure ftp://ftp.flybase.net/releases/current/precomputed_files/genes/

Apps GitHub Reactome 16 Calendar Feulgen Docs DataViz Coronavirus #s FC

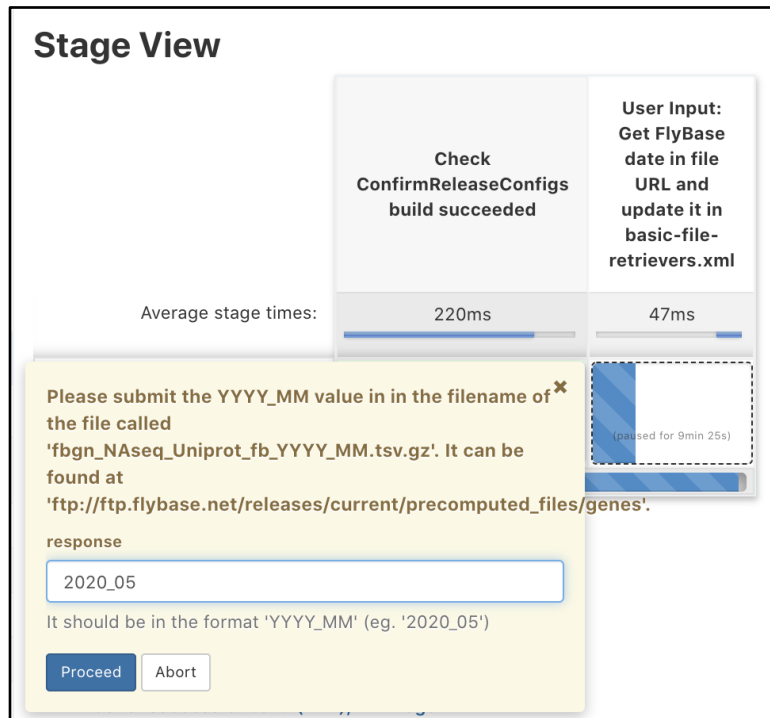
Index of /releases/current/precomputed_files/genes/

[parent directory]

Name	Size	Date Modified
automated_gene_summaries.tsv.gz	5.5 MB	10/3/20, 2:20:00 PM
dmel_unique_protein_isoforms_fb_2020_05.tsv.gz	164 kB	9/28/20, 11:26:00 AM
fbgn_NAseq_Uniprot_fb_2020_05.tsv.gz	7.5 MB	9/28/20, 11:21:00 AM
fbgn_annotation_ID.tsv.gz	0 B	10/12/20, 2:24:00 PM
fbgn_annotation_ID_fb_2020_05.tsv.gz	233 kB	9/28/20, 11:20:00 AM
fbgn_exons2affy1_overlaps.tsv.gz	620 kB	10/3/20, 1:04:00 PM
fbgn_exons2affy2_overlaps.tsv.gz	690 kB	10/3/20, 1:04:00 PM
fbgn_fbtr_fbpp_expanded_fb_2020_05.tsv.gz	694 kB	9/28/20, 11:19:00 AM
fbgn_fbtr_fbpp_fb_2020_05.tsv.gz	259 kB	9/28/20, 11:18:00 AM
fbgn_gleanr_fb_2020_05.tsv.gz	1.4 MB	9/28/20, 11:20:00 AM
gene_functional_complementation_fb_2020_05.tsv.gz	6.3 kB	9/28/20, 11:22:00 AM
gene_genetic_interactions_fb_2020_05.tsv.gz	268 kB	9/28/20, 7:09:00 AM
gene_group_data_fb_2020_05.tsv.gz	129 kB	9/28/20, 11:22:00 AM
gene_groups_HGNC_fb_2020_05.tsv.gz	22.8 kB	9/28/20, 11:22:00 AM
gene_map_table_fb_2020_05.tsv.gz	1.7 MB	9/28/20, 11:27:00 AM
gene_rpk_matrix_fb_2020_05.tsv.gz	1.6 MB	9/28/20, 11:31:00 AM
gene_rpk_report_fb_2020_05.tsv.gz	11.0 MB	9/28/20, 11:46:00 AM
gene_snapshots_fb_2020_05.tsv.gz	399 kB	9/28/20, 11:23:00 AM
ncRNA_genes_fb_2020_05.json.gz	1.2 MB	9/28/20, 11:19:00 AM
physical_interactions_mitab_fb_2020_05.tsv.gz	2.9 MB	9/28/20, 11:54:00 AM

Remove first list from the guide to pharmacology files which starts with "#

Once you have that value, submit it in the answer field of the Jenkins question and click 'Proceed':



After this, Jenkins will run to completion, pausing before the 'archive' step so that the Release runner can review the logs and files downloaded.

Validation:

After the downloads program has run, Jenkins should have sent an email notifying the developer. This is necessary because Jenkins 'pauses' the step to allow the developer to review the logs that are on the Release server before uploading them to S3.

There are a few ways to verify that AddLinks downloaded all files correctly. One of the main challenges with validating this step is that there are a lot of files downloaded, and so comparing between releases is useful, but is time intensive and it is easier to just see if any file downloads didn't complete, and to see the size of the files that were downloaded.

- 1) On the release server, navigate to `/var/lib/jenkins/workspace/Releases/XX/AddLinks-Download/logs/retrievers`. Check the 'tail' of all files sitting in this directory('tail *'). There is still going to be a lot of output, but this allows you to somewhat quickly review how the individual downloaders for the step ran. When a file downloader finishes correctly, it should have a message that says something like:

'File Info: Name: `/tmp/addlinks-downloaded-files/downloadedFile`, **Size: 657110**, Created: 2020-11-19T22:52:36Z, Modified: 2020-11-19T22:52:36Z'.

What you want to see is that there is a non-zero 'size' value for each of the downloaders.

You can also use `'tail * | grep 'ERROR''` or `'grep -Hrn 'ERROR''` to try and get a concise version of any error messages while in the 'AddLinks-Download/logs/retrievers/' folder. You might get a lot of errors reported in the 'EnsemblBioMart', 'UniProt' and 'KEGGRetriever' log files. At time of writing the EnsemblBioMart and UniProt ones could be ignored, but as I (Justin) am not the author of KEGGRetriever, I can't say whether they can be ignored or not.

- 2) Even if some files don't have that 'File Info' string, it doesn't necessarily mean that the download failed. For 'downloader' logs that are suspect, look in the log file for evidence of the file name/path. Once you have that, navigate to the '/tmp/addlinks-downloaded-files/' directory and look for evidence of that file. Confirm it has a non-zero size. You can also take this time to see if there are any 0-byte files in this directory. Beware that gzipped 0-byte files will have a non-zero file size, usually hundreds of bytes in size.
- 3) As a final check, you can review the console logs from the AddLinks-Download. Be sure to view the 'Full Log' instead of the truncated logs that are visible by default. Look for any evidence of improper AddLinks behavior, which should be reported as Exception messages in the console logs.
 - a) **Note:** At time of writing, there are many 'EnsemblBioMartRetrieverBioMartQueryException' messages being thrown due to some species not having available BioMart data. This is OK. It is an artifact from the old *OtherIdentifiers* step, that has been merged with AddLinks. It just reflects that some species-specific data is not available in BioMart. As a sanity check to ensure that this step ran though, navigate to '/tmp/addlinks-downloaded-files/biomart' and make sure there are many - comparable to last release - '\${species}_microarray_go_ncbi_ids' and '\${species}_uniprot' files.
 - b) If you notice output from the KeggFileRetriever that says something like 'Sorry, No uniprot-to-kegg mappings found for species \${species name} / \${db identifier}', then check if a file that contains that identifier in its name exists in '/tmp/addlinks-downloaded-files/kegg_entries' in the format 'kegg_entries.\${db identifier}.2.txt'. For reference, the '2' here is also a Reactome database identifier, and it corresponds to the 'UniProt' *ReferenceDatabase* instance in the database, while the other one corresponds to the described species instance. [Insert explanation about updating config file/branch and re-running]
 - c) TargetPathogen is known to not find the file. The website no longer appears to provide us with the necessary file and doesn't appear to be maintained.

Once you're satisfied with the AddLinks file downloads, navigate back to the AddLinks-Download page and click 'Proceed' in the 'User Input: Confirm successful AddLinks file downloads' pop up. Jenkins will then archive the AddLinks-Download run, including the downloaded Addlinks files, on the Reactome S3 bucket. For completeness, you can navigate to

'S3://reactome/private/releases/XX/add_links/downloads/data/' and confirm that 'addlinks-downloads-vXX.tgz' is present.

AddLinks-Insertion (8 hrs)

AddLinks-Insertion requires that the AddLinks-Download step has been successfully created for it to run.

Try to ensure that there are at least 40 GB free to run this. You may need more, if the size of the input files grows (and it probably will, over time).

The AddLinks-Insertion step takes all the files downloaded during the AddLinks-Download step and creates cross-references from the identifiers found in those files, and also adds in microarray, GO, and NCBI Gene IDs to the 'otherIdentifier' attributes of ReferenceEntity instances. These cross-references are used to create link-outs from the Reactome website to a number of other bioinformatic resources while the 'otherIdentifier' values are used by the Reactome Analysis Service.

Navigate to the AddLinks-Insertion step page in Jenkins at '*Dashboard > Releases > XX > Database Updates*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation:

Most of AddLinks-Insertion can be evaluated for success just by looking at the console logs. AddLinks-Insertion can be broadly categorized into the 'File processing' and 'Reference creation' steps. The former just entails the process of creating identifier maps from the data files downloaded during AddLinks-Download. The mappings are generally of the structure {UniProtIdentifier:DataFileIdentifier}, as UniProt is the primary database that Reactome maps to.

AddLinks is a complex process that needs to be validated thoroughly. Below are 4 ways to check the step ran successfully. I highly recommend you follow these instructions as this (along with Orthoinference) is one of the critical 'time-intensive' Release steps, and so knowing that it ran successfully will make your life easier should you need to re-run some steps during a Release.

- 1) Now you will want to evaluate the link-checking performed by Addlinks. With the full console logs opened, search for the phrase 'links were OK'. You should see something like 'X \${ReferenceDatabase} links were OK, Y links were NOT ok'. There should be a lot of positive hits for this phrase if Addlinks ran successfully. As long as there are no 'NOT ok' links, then you can proceed to the next one.

Some servers will fail for a small handful of links. Usually this happens if a server just doesn't respond fast enough. It's worth checking one or two of the failures to see if they are still a problem, or if it was just a transient issue while link-checking was running.

If ALL links for a resource fail, it's worth looking into that.

Note: At time of writing, the Monarch link-checking returns false-positives. The pages resolve, it's just that the link-checking is done by scraping pages for identifiers, and Monarch actually loads the page after you arrive, meaning that a web scraper couldn't pick up any of the content.

From the database you can do queries like:

You should be able to find an instance with all 3 types:

- 1) GO is always prefixed with the 'GO:' string.
 - a) Query: "SELECT * FROM ReferenceEntity_2_otherIdentifier WHERE otherIdentifier LIKE "GO:%" LIMIT 20;"
- 2) Microarray
 - a) Query: "SELECT * FROM ReferenceEntity_2_otherIdentifier WHERE otherIdentifier LIKE 'GgaAffx%' LIMIT 20;"
- 3) If present, the NCBI identifier can be submitted into the search form at [NCBI Gene](#). The returned page should correspond to the name of the instance you took it from. For example, the above NCBI identifier, 423618, returns a page with the gene 'OGDHL'. This corresponds to the name of the instance in the CuratorTool, which is 'ENSEMBL:ENSGALP00000003546 **OGDHL**'.
 - a) Query "SELECT * FROM ReferenceEntity_2_otherIdentifier WHERE otherIdentifier LIKE "ENSGALP%" LIMIT 20;"
- 4) The microarray identifiers are highly variable, as they correspond to a number of different microarray types. If you see identifiers that aren't just a string of numbers (which would be NCBI), or prefixed with 'GO:', then they are likely microarray identifiers. You just want to confirm that some instances have these third, complex-looking identifier types.

- Look at add-links-checker step to see if there are any blockers

If all those checks passed, then you are OK to proceed to the next step!

BioModels (4-5 hrs)

BioModels is the final step in the 'Database Updates' phase. It is a bit unusual in that it actually requires an intermediate graph database and analysis core to be produced before it can actually be run. It requires GenerateGraphDatabase

As described above, BioModels will first create a graph-database from the *release_current* database, and then create an intermediate analysis.bin file from this graph database. After it does this, it then runs a biomodels-mapper to produce the 'models2pathways.tsv' file. This is actually the first of the files that needs to be added to the 'downloads' directory. Once all of those things have been done, the relational database updates finally take place. Using the 'models2pathways.tsv' file that was just produced, a Java program is run that creates BioModels cross-references in the database.

Navigate to the BioModels step page in Jenkins at '*Dashboard > Releases > XX > Relational-Database-Updates*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

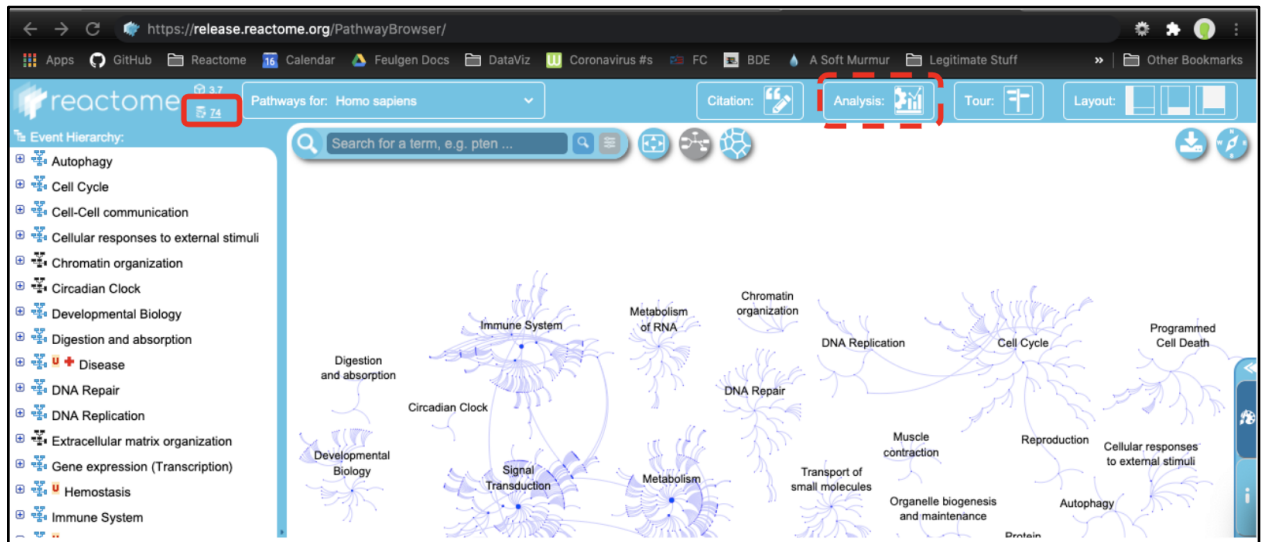
Validation

This step is relatively easy to validate, as you only need to confirm that the analysis bin file was properly created and set up, that the 'models2pathways.tsv' file looks correct and is placed in the correct location, and that BioModels cross-references were created in *release_current*

- 1) Navigate to '/usr/local/reactomes/Reactome/production/AnalysisService/input' on the release server and confirm that the 'analysis.bin' symlink is to the recently created 'analysis-biomodels-vXX.bin' file.
- 2) Navigate to '/usr/local/reactomes/Reactome/production/Website/static/download/XX' and confirm that the file 'models2pathways.tsv' is present. You can compare this to the previous release to see if it looks like the correct size. If the 'download/XX-1' directory is still present, you should be able to find the previous releases' 'models2pathways.tsv' file there. As long as the files look similar in size and structure, you should be good to proceed.
- 3) Finally, we just want to confirm that the BioModels cross-references have been added to the database. In either MySQL or a release-server-connected CuratorTool, find the number of 'DatabaseIdentifier' instances that contain 'BioModels' (including capitalized M) in their display name. At time of writing, there should be more than 100 instances. You can also confirm that a ReferenceDatabase instance with displayName 'BioModels Database' exists.
 - a) Using SQL:
 - i)

```
SELECT count(*) from DatabaseObject WHERE DatabaseObject._class = 'DatabaseIdentifier' AND DatabaseObject._displayName LIKE '%BioModels%'
```

- 4) Navigate to '/usr/local/reactomes/Reactome/production/AnalysisService/input' on the release server and confirm that the 'analysis.bin' symlink is to the recently created 'analysis_vXX.bin' file.
- 5) In a browser, navigate to <https://release.reactome.org/PathwayBrowser/> and confirm that there are no warning flags visible under the 'Analysis' tab (red dashed box):



Below is what the Analysis tab will look like if there is an issue. It will be yellow/orange if the Analysis.bin file exists, but does not match the graph database checksums. It will be red if the file can't be found.



If the Analysis tab has a warning flag (as above), chances are something went wrong and that the current graph database is not the same database that was used to generate the currently symlinked analysis bin file. Examine the following two URLs:

<https://release.reactome.org/ContentService/data/database/info>

<https://release.reactome.org/AnalysisService/database/info>

The ContentService URL gives information on the graph database, while the AnalysisService URL gives information on the Analysis.bin file that is currently being used. They should be the same. If they aren't, you can use the mismatched values to help determine where things have gone wrong. Most often the Analysis.bin file wasn't actually generated, sym-linked, or tomcat wasn't restarted.

Statistics generator (2mins)

Click play with the statistics-generator pipeline. The files will get uploaded to S3 (download.reactome.org/<version>/stats/)

Validation

Look on the release.reactome.org:

- [Homepage stats](#)
- <https://release.reactome.org/about/statistics>
- <https://release.reactome.org/documentation/inferred-events>

File-Generation

The File-Generation stage consists of 8 steps, all of which primarily produce files that are consumed by Reactomes users. The sole exception is the *DiagramConverter* step which, aside from producing diagram files, also makes a final set of modifications to the graph database. The 8 steps are *DiagramConverter*, *FireworksLayout*, *DataExport*, *SBMLExporter*, *InteractionExporter*, *DiagramExporter*, *EventPDF*, and *DownloadDirectory*. These steps generally run in less than an hour and all populate the `/usr/local/reactomes/Reactome/production/Website/static/download/XX` folder with data files.

DiagramConverter (30 mins)

This step is notable for being the final step that actually modifies any database in Reactome's Release pipeline. After this step completes, the final 'reactome.graphdb.tgz' file is produced and added to the downloads/XX folder, along with the 'diagram' folder.

Navigate to the DiagramConverter step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Zip diagram folder:

1. Run `cd /usr/local/reactomes/Reactome/production/Website/static/download/current/`
2. Run `tar -czf diagram.tgz diagram`

Validation

There are a few items that need to be completed to ensure that DiagramConverter ran correctly. This step is responsible for both the final graph database and Pathway diagrams in Reactome that can be seen in the Pathway Browser.

- 1) Check the output for the 'Compare previous release file number' step. The DiagramConverter should have produced thousands of json files. At time of writing, it was generally over 30k files, and so the output for this step is expected to be similar to that number.
- 2) Check the /usr/local/reactomes/Reactome/production/Website/static/download/XX/ folder for the 'diagram' folder, which should be filled with the json files mentioned in step 1.
- 3) Check the /usr/local/reactomes/Reactome/production/Website/static/download/XX/ folder for the 'reactome.graphdb.tgz' file.

Fireworks Layout (5 mins)

FireworksLayout generates the JSON files that are used for the Reactome fireworks view in the Pathway Browser. A fireworks file is produced for all species in Reactome.

Navigate to the FireworksLayout step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

- 1) Check the output from the 'Compare file sizes with previous release' stage of FireworksLayout. There should be the same number of json files between Releases, and they should all be named after an organism (eg: Mus_musculus.json). These files should be close in size, though it is common for the files to be a bit bigger for the current Release.
- 2) Confirm the 'fireworks' folder exists in /usr/local/reactomes/Reactome/production/Website/static/download/XX.

Exporters

Data Exporters (40 mins)

This step produces a lot of data files that appear on Reactome's [downloads page](#). They are mostly mapping files from Reactome identifiers to external resource identifiers. One thing to note about this step is that it uses a lot of RAM. At time of writing the step gets by temporarily shutting down **tomcat9** and **mysql**, while having the Java memory ceiling set to 16000 MB. As more data gets added to Reactome, it's possible that at some future date the release server will not be able to run this step without increasing the amount of available memory. Something to keep in mind!

Navigate to the DataExport step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

- 1) Review the output from the 'Compare Data-Export file line counts between releases' step of DataExport. This should show the number of files between releases, and compare the line counts between them. You may notice some files differ by thousands of lines, but keep in mind that some of these files have over a million lines! As long as there wasn't a drop off in lines over 10%, chances are the files are fine.
- 2) At time of writing, the file ``reactome_reaction_exporter_vXX.txt`` wasn't being output by the step described in stage 1, as it compares file names between the two releases. This file has the version number in its name, and the comparison isn't robust enough to handle that. This file will need to be compared between releases manually. You should be able to find both of them in the corresponding `/usr/local/reactomes/Reactome/production/Website/static/download/` folder to their release.
- 3) Ensure that the export files were moved to `/usr/local/reactomes/Reactome/production/Website/static/download/XX`. You should see a large number (approx. 50) of txt files in this folder. Below is the list of files. At time of writing, there were 52 produced by the DataExport step:
 - ChEBI2Reactome.txt
 - ChEBI2ReactomeReactions.txt
 - ChEBI2Reactome_All_Levels.txt
 - ChEBI2Reactome_PE_All_Levels.txt
 - ChEBI2Reactome_PE_Pathway.txt
 - ChEBI2Reactome_PE_Reactions.txt
 - ComplexParticipantsPubMedIdentifiers_human.txt
 - Complex_2_Pathway_human.txt
 - Ensembl2Reactome.txt
 - Ensembl2ReactomeReactions.txt
 - Ensembl2Reactome_All_Levels.txt
 - Ensembl2Reactome_PE_All_Levels.txt
 - Ensembl2Reactome_PE_Pathway.txt
 - Ensembl2Reactome_PE_Reactions.txt
 - Ewas2Pathway_human.txt
 - HumanDiseasePathways.txt
 - GtoP2Reactome.txt
 - GtoP2ReactomeReactions.txt
 - GtoP2Reactome_All_Levels.txt
 - GtoP2Reactome_PE_All_Levels.txt

- GtoP2Reactome_PE_Pathway.txt
- GtoP2Reactome_PE_Reactions.txt
- IntAct_Static.txt
- LowerLevelPathway2Topic.txt
- LowerLevelPathway2Topic_Human.txt
- NCBI2Reactome.txt
- NCBI2ReactomeReactions.txt
- NCBI2Reactome_All_Levels.txt
- NCBI2Reactome_PE_All_Levels.txt
- NCBI2Reactome_PE_Pathway.txt
- NCBI2Reactome_PE_Reactions.txt
- Pathways2GoTerms_human.txt (**new as of Release v90 - *check for existence and appropriate size, then message people from GO when it is ready:**
<https://github.com/geneontology/go-ontology/issues/27081>)
- ProteinRoleReaction.txt
- ReactionPMIDS.txt
- Reactions2GoTerms_human.txt
- Reactome2OMIM.txt
- ReactomePathways.txt
- ReactomePathwaysRelation.txt
- UniProt2Reactome.txt
- UniProt2ReactomeReactions.txt
- UniProt2Reactome_All_Levels.txt
- UniProt2Reactome_PE_All_Levels.txt
- UniProt2Reactome_PE_Pathway.txt
- UniProt2Reactome_PE_Reactions.txt
- UniProt_Entity_LiteratureReferences.txt
- miRBase2Reactome.txt
- miRBase2ReactomeReactions.txt
- miRBase2Reactome_All_Levels.txt
- miRBase2Reactome_PE_All_Levels.txt
- miRBase2Reactome_PE_Pathway.txt
- miRBase2Reactome_PE_Reactions.txt
- reactome_reaction_exporter_vXX.txt

SBML Exporter (4 hrs)

This step generates all SBML files that can be downloaded within the Reactome Pathway Browser.

Navigate to the SBMLExporter step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

This step is pretty simple to validate. All you need to do is confirm the presence of two files, 'all_species.3.1.sbml.tgz' and 'homo_sapiens.3.1.sbml.tgz' in /usr/local/reactomes/Reactome/production/Website/static/download/XX, and confirm that they appear to be similar in size to the previous releases' version of these files. The previous release files can be found in either /usr/local/reactomes/Reactome/production/Website/static/download, or on S3.

Interaction Exporter (45 mins)

InteractionExporter produces dump files of Reactome interaction data that can be downloaded from Reactome's [downloads page](#). It produces 4 files, 2 of which are Human-specific, and 2 that pertain to all species interaction data in Reactome.

Navigate to the InteractionExporter step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

This step can be validated pretty simply. Look for an 'interactors' folder in /usr/local/reactomes/Reactome/production/Website/static/download/XX. Within this folder there should be four files:

- reactome.homo_sapiens.interactions.psi-mitab.txt
- reactome.homo_sapiens.interactions.tab-delimited.txt
- reactome.all_species.interactions.psi-mitab.txt
- reactome.all_species.interactions.tab-delimited.txt

If all four of those files are present, and if they look similar in size (if not a bit larger) to those from the previous release, then you can proceed to the next step. The previous release files can be found in either /usr/local/reactomes/Reactome/production/Website/static/download, or on S3.

Diagram Exporter (20 mins)

The DiagramExporter step creates three archives that contain the Reactome Pathway diagrams in three different formats, SVG, PNG, and SBGN. The archives, 'diagrams.svg.tgz', 'diagrams.png.tgz', and 'homo_sapiens.sbgn.tar.gz', are all found on the Reactome [downloads page](#).

Make sure that the ehld directory and svgsummary.txt files exist. These should be in /usr/local/reactomes/Reactome/production/Website/static/download/XX/ehld

They are placed there by Cris. If they are not available, THIS STEP WILL FAIL.

Navigate to the DiagramExporter step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

This is another step that is relatively easy to verify. Confirm that the three files are present in `/usr/local/reactomes/Reactome/production/Website/static/download/XX`:

- `diagrams.svg.tgz`
- `diagrams.png.tgz`
- `homo_sapiens.sbgm.tar.gz`
- `humanPathwaysWithDiagrams.txt`
- `pathway2summation.txt`

You can also compare the size of these archives with the previous release. The previous release files can be found in either `/usr/local/reactomes/Reactome/production/Website/static/download`, or on S3.

Finally, you can also get a count of the number of files in the archive using the following command:5

```
tar -tvf ${file} | wc -l
```

If you use this command on each of the DiagramExporter archives in the `download/XX/` and `download/XX-1`, you will see the number of files that were produced in each release. At time of writing, there were generally over a 1200 in a proper run.

Event PDF (30 mins)

EventPDF produces 'TheReactomeBook.pdf.tgz' archive that can be downloaded from Reactome's [downloads page](#). This archive contains PDF 'books' that are organized by Reactome's TopLevelPathways (eg: 'Cell Cycle'). You can find the whole of Reactome's Human Curations within this data file.

Navigate to the DiagramExporter step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

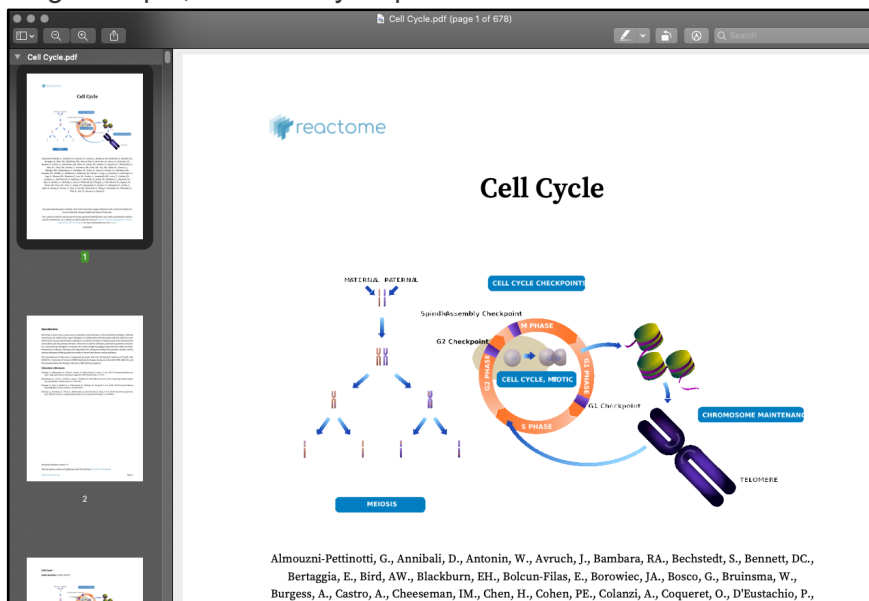
There are a few things that need to be done to verify that the EventPDF step ran correctly. As usual, you will need to check that the file's contents look correct, and that the archive was

moved to the downloads/XX folder, but you will also want to confirm that some of the PDF files actually render correctly.

- 1) Navigate to the console logs for the 'Compare TheReactomeBook contents between releases'. This step will output the contents of 'TheReactomeBook.pdf.tgz' archives from the current and previous releases. Confirm that the output from this step looks correct by looking at file sizes and that the same TopLevelPathway.pdf files exist. There should not be a change in the number of files, unless Curation has added or removed a TopLevelPathway for this release.
- 2) Confirm that the 'TheReactomeBook.pdf.tgz' file is present in /usr/local/reactomes/Reactome/production/Website/static/download/XX.
- 3) To verify that the PDFs render properly, you will actually need to bring the file to your local computer. You can use scp the file over or download the file from S3. Once you have the archive, you can unpack it using the following command:

```
tar -xf TheReactomeBook.pdf.tgz
```

Once the archive has been unpacked, open any of the PDF files in the 'TheReactomeBook' folder that should be visible. Once opened, the first page should have a diagram (probably an EHLN one) and title corresponding to the file you opened. In the following example, the 'Cell Cycle.pdf' file was reviewed:

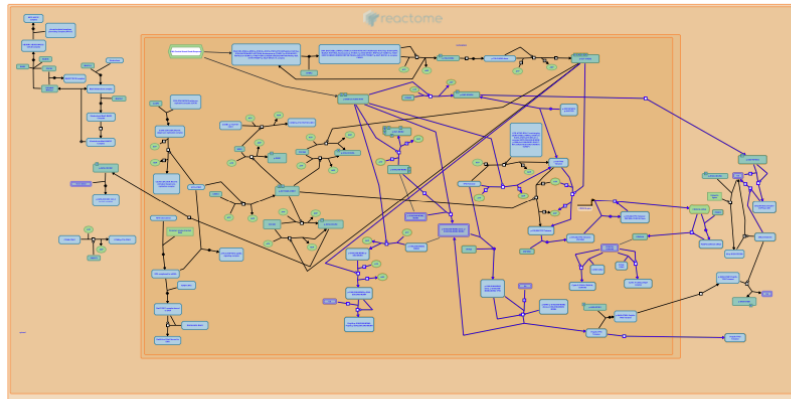


Scroll through a few of the PDF pages. You should see Pathway/Reaction diagrams (EHLN and regular diagrams), text describing those diagrams, citations, and editing histories:

p53-Dependent G1/S DNA damage checkpoint [↗](#)

Location: [Cell Cycle](#) → [Cell Cycle Checkpoints](#) → [G1/S DNA Damage Checkpoints](#)

Stable identifier: R-HSA-69580



The arrest at G1/S checkpoint is mediated by the action of a widely known tumor suppressor protein, p53. Loss of p53 functions, as a result of mutations in cancer prevent the G1/S checkpoint (Kuerbitz et al, 1992). P53 is rapidly induced in response to damaged DNA. A number of kinases, phosphatases, histone acetylases and ubiquitin-conjugating enzymes regulate the stability as well as transcriptional activity of p53 after DNA damage.

Literature references

Kuerbitz, SJ., Plunkett, BS., Walsh, WV., Kastan, MB. (1992). Wild-type p53 is a cell cycle checkpoint determinant following irradiation. *Proc Natl Acad Sci U S A*, 89, 7491-5. [↗](#)

Editions

2003-06-05	Authored	Khanna, KK.
2020-09-12	Edited	Joshi-Tope, G.

Once you are satisfied this PDF has been populated correctly, we just need to check a few more things - the table of contents (TOC), embedded links, and the release version. First, the TOC. Strangely, the TOC is actually found at the end of the PDF document. Navigate to the end of the document, and you should see something like the following:

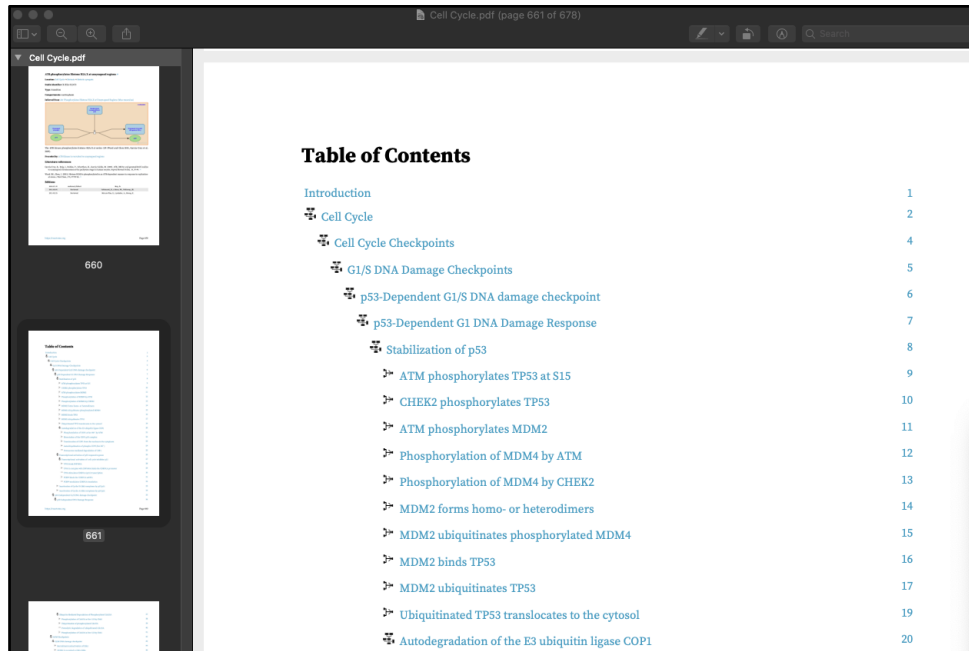
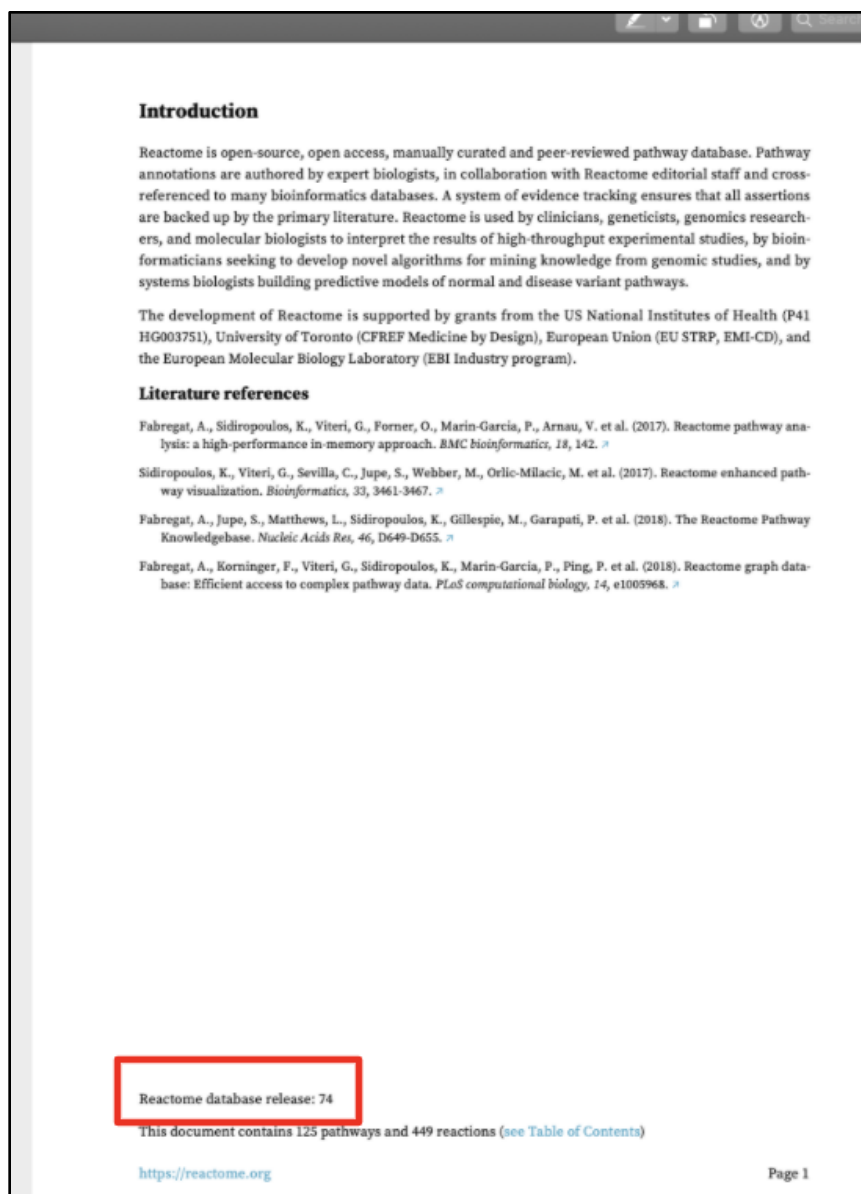


Table of Contents	
Introduction	1
Cell Cycle	2
Cell Cycle Checkpoints	4
G1/S DNA Damage Checkpoints	5
p53-Dependent G1/S DNA damage checkpoint	6
p53-Dependent G1 DNA Damage Response	7
Stabilization of p53	8
ATM phosphorylates TP53 at S15	9
CHEK2 phosphorylates TP53	10
ATM phosphorylates MDM2	11
Phosphorylation of MDM4 by ATM	12
Phosphorylation of MDM4 by CHEK2	13
MDM2 forms homo- or heterodimers	14
MDM2 ubiquitinates phosphorylated MDM4	15
MDM2 binds TP53	16
MDM2 ubiquitinates TP53	17
Ubiquitinated TP53 translocates to the cytosol	19
Autodegradation of the E3 ubiquitin ligase COP1	20

Click on the 'Introduction' text. This should bring you to a page that looks like the below image. At the bottom of this page is a line that should say 'Reactome database release: XX' (red box):



Finally, we just want to make sure the embedded links work for the Pathways/Reaction pages as well. Click 'see Table of Contents' on this Introduction page. From here, just click any of the Pathway/Reactions listed and confirm that the embedded URL brings you to the corresponding page.

If all of that worked as expected, then great - you are finished EventPDF!

Download Directory (3 hrs)

DownloadDirectory is another step that populates the Reactome [downloads page](#). This step focuses on files that are created in particular formats, such as 'GO Annotation' and 'BioPAX'

files. These files are actually created from the relational database, so its only real prerequisite is that the Relational-Database-Updates have been completed.

Before running DownloadDirectory, make sure that the following properties are available in Jenkins config file. Config.properties file from this repo is not used for this pipeline.

- protegeexporter.parallelism=5
- protegeexporter.filterSpecies=Homo sapiens
- protegeexporter.pathToWrapperScript=/var/lib/jenkins/workspace/Releases/XX/File-Generation/DownloadDirectory/src/main/resources/
 - Be sure to update the XX value above

Note: The 'Adding or Updating the Master Configuration File' section can be found near the beginning of this document.

Navigate to the DownloadDirectory step page in Jenkins at '*Dashboard > Releases > XX > File-Generation*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

If it needs to be re-run and those files don't need to be made, it is best to turn off those steps using the 'stepsToRun.config' file found in the resources folder of the [release-download-directory](#) repository.

Validation

Confirm that the following files/folders are visible in the /usr/local/reactomes/Reactome/production/Website/static/download/XX folder:

- ReactomePathways.gmt.zip
- databases/
 - gk_stable_ids.sql.gz
 - gk_current.sql.gz
- protege_files.tar.gz
- reactome_data_model.pont
- reactome_stable_ids.txt
- ReactomeToBioSystems.zip
- biopax2.zip
- gene_association.reactome.gz
- reactome_data_model.pins
- reactome_data_model.pprj

Compare each of these file sizes between releases. As long as they appear to be somewhat close, you should be good to proceed. Pay particular attention to protege_files.tar, as an empty 'tar' file still has over 1000 bytes, which can be missed at a glance.

BioPax Exporter (4hrs)

Go to Dashboard > Release > XX > File-Generation > BioPax3Exporter and run the pipeline

This will create a biopax.zip file and move it into the download directory.

GOCAM (3hrs 15 mins)

Run "Dashboard > Release > XX > GOCAM"

Validation:

- Ensure there is enough hard drive space (at least 19G)
- There should be a gocam.zip file added to the download directory
- Check there are approximately the same number of files inside as the last release and that it has about the same size

Search Indexer (7.5 hrs)

This step generates the Solr search indexing that is used on the Reactome website. It produces a variety of files including the ebeye, icons and sitemap files.

Navigate to the SearchIndexer step page in Jenkins at '*Dashboard > Releases > XX*'. If you can't see the *Build Now* option on the left-hand side of the steps' page, then you will need to select *Configure* on the left-hand side and then click *Save* at the bottom. Select *Build Now* from the main page if it's visible.

Validation

There are two things that need to be checked to make sure that the search indexer ran correctly. You will need to check the /usr/local/reactomes/Reactome/production/Website/static/download/XX folder for the following files/folders:

- ebeye.xml.gz
- ebeyecovid.xml.gz
- icons/ (23 files)
 - UNIPROT2Icon.txt
 - PFAM2Icon.txt
 - NCBI2Icon.txt
 - GO2Icon.txt
 - CL2Icon.txt
 - UBERON2Icon.txt
 - OPL2Icon.txt
 - MESH2Icon.txt
 - ENSEMBL2Icon.txt

- CHEBI2Icon.txt
- SO2Icon.txt
- OMIT2Icon.txt
- KEGG2Icon.txt
- ENA2Icon.txt
- BTO2Icon.txt
- RFAM2Icon.txt
- NP2Icon.txt
- IUPHAR2Icon.txt
- DOID2Icon.txt
- PUBCHEM2Icon.txt
- NCIT2Icon.txt
- INTERPRO2Icon.txt
- COMPLEXPORTAL2Icon.txt

You will also need to check that the site map files were updated in the Website folder using the following commands:

```
ls -l /usr/local/reactomes/Reactome/production/Website/static/sitemapindex.xml
```

```
ls -l /usr/local/reactomes/Reactome/production/Website/static/sitemap/*
```

Confirm that each of the three files output by these two commands (sitemapindex.xml, sitemap_1.txt.gz, sitemap_2.txt.gz) were created around the time of the Jenkins SearchIndexer build, and confirm that their owner:group is set to 'www-data:reactome'

If all of those items look correct, then you can proceed to the next step (if there are any).

Upload docker image to AWS ECR

Make tags:

```
docker tag reactome/graphdb:ReleaseXX public.ecr.aws/reactome/graphdb:ReleaseXX
```

```
docker tag public.ecr.aws/reactome/graphdb:ReleaseXX
```

```
public.ecr.aws/reactome/graphdb:latest
```

Run these commands **logged in as jenkins user**:

- `aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws`
- `docker push public.ecr.aws/reactome/graphdb:ReleaseXX`
- `docker push public.ecr.aws/reactome/graphdb:latest`

Finalize Release

- Run Finalize Release pipeline
- Email curators list saying that it is ready for review (Lisa and Karen)

Release database testing

- See the [Release Database Testing document](#) (*Appendix R3*)

Pass2Production -P2P (1.5 hr)

Before starting P2P, take a snapshot of DEV and PROD via AWS.

Pass2Production is run entirely through the 'pass2production.sh' script found at **release.reactome.org:/usr/local/reactomes/Reactome/production/Website/scripts/**. It is run 4 times, twice (move_data_over and do_release) for DEV and twice for PROD.

You will need the following:

- 'Shared' account (user is /home/shared) credentials - this is an account that is used by multiple users so that anyone running P2P can run this script without needing to share their own certificates or passwords.
- 'Joomla' database credentials
- MySQL 'curator' credentials
- Release 'sudo' credentials

Once you have those credentials, you should be good to proceed with P2P.

- 1) Email Reactome internal announcing that you will be starting the test run of pass2production on DEV.
- 2) Run "cd /usr/local/reactomes/Reactome/production/Website/scripts"
- 3) Check to make sure there are no differences between the scripts:
 - a) Run "diff ../static/management/pass2production.sh ./pass2production.sh"
 - b) If returns nothing you are good to go
- 4) **Make sure there is enough space on dev (>18Gbs)**

- 5) Run the following '**move_data_over**' p2p command for **DEV**, using the appropriate arguments, and answering all prompts that appear:
 - Run "sudo ./pass2production.sh release_version=XX destination_server=DEV email_me=reactome-developer@reactome.org move_data_over"
 - Once this finishes, you should receive an email that matches the output from the p2p script. Review either one of those for WARN or ERROR messages.
- 6) Now run the '**do_release**' p2p command for **DEV**, answering all prompts:
 - Run "sudo ./pass2production.sh release_version=XX destination_server=DEV email_me=yourEmailAddress do_release"
 - Review the output/email contents again, looking for WARN or ERROR messages.
- 7) Once this has run, you will need to check the DEV version of the website, dev.reactome.org. See the 'Items to check' list after these instructions for items that should be checked.
- 8) Assuming everything looks correctly updated on DEV, it is now time to do the same p2p commands for PROD. Send an email to Reactome internal to let them know that you are running P2P for PROD & let the testers know that testing can be done on DEV.
- 9) Check that it would using this list "Items to check after completing p2p" below
- 10) Do load testing on dev
- 11) **Make sure there is at least 18 GBs available on PROD!!!**
- 12) Write email to internal list saying we are releasing to production
- 13) Run the '**move_data_over**' p2p command for **PROD**, answering all prompts:
 - Run "sudo ./pass2production.sh release_version=XX destination_server=PROD email_me=reactome-developer@reactome.org move_data_over"
 - Review email/output for WARN or ERROR messages.
- 14) Run the '**do_release**' p2p command for **PROD**, answering all prompts:
 - Run "sudo ./pass2production.sh release_version=XX destination_server=PROD email_me=yourEmailAddress do_release"
 - Review email/output for WARN or ERROR messages.
- 15) Once the do_release for PROD has finished, you have officially completed Reactome's *release!* But your work isn't finished yet. Review the items on the 'Items to check' list.
- 16) When you are satisfied with the state of the website, send a final email to Reactome internal (reactome-internal@oicr.on.ca) to let the team know that the release was successful.

Items to check after completing p2p:

- Checksums for Analysis and Content should be the same:
 - DEV:
 - <https://dev.reactome.org/ContentService/data/database/info>
 - <https://dev.reactome.org/AnalysisService/database/info>
 - PROD:
 - <https://reactome.org/ContentService/data/database/info>
 - <https://reactome.org/AnalysisService/database/info>

- Pathway Browser
 - DB version should be correct number
 - No red/yellow flags should be visible on Analysis tab
 - Try navigating through some of the hierarchy
 - Try navigating through an 'Inferred' pathway
- Analysis Service
 - Run a 'UniProt accession list' analysis
 - Run a 'Microarray data' analysis
 - Run any analysis with 'interactors'
- TOC/DOI pages should load, sometimes taking a minute or so. Look for 'New' or 'Updated' values matching the releaseDate parameter from the configuration file. The TOC page might not have anything new since it relates to TopLevelPathways, but the DOI page should have new annotations.
 - DEV:
 - <https://dev.reactome.org/content/toc>
 - <https://dev.reactome.org/content/doi>
 - PROD:
 - <https://reactome.org/content/toc>
 - <https://reactome.org/content/doi>
- Find 'NEW' DOI from the DOI page and use its name to test search
 - Review its Details page
- Navigate to the PathwayBrowser of that pathway
- Check citations tool
- Click around and test downloads/urls in PathwayBrowser
- Go to downloads page and test that some are working
 - Compare /download/current directorios between production and release
 - DEV: <https://dev.reactome.org/download/current/>
 - PROD: <https://reactome.org/download/current/>
 - RELEASE: <https://release.reactome.org/download/current/>
- Go to news and check it has new release information
- Go to stats and see that it is for the new release

Post-Release

Upload to Zenodo

- Download directory
 - Get stats folder in the download directory
 - Switch to s3bot user

- RUN "cd /usr/local/reactomes/Reactome/production/Website/static/download/<version>"
- RUN "mkdir stats"
- RUN "cd stats"
- RUN "aws s3 sync s3://download.reactome.org/<version>/stats/ ."
- Gzip download directory
 - RUN "zip -r reactome-download-directory.zip <download-directory>"
- Download locally
 - RUN "scp reactome-release:/usr/local/reactomes/Reactome/production/Website/static/download/reactome-download-directory.zip ."
- Go to Zenodo interface and add a new version
 - Will need to change version number (e.g. v90) and the publication date (e.g. 2024-09) and then publish
- Documentation
 - This is done whenever there are significant changes
 - aim for yearly
 - Take files from s3 and remove the version in the names and upload to Zenodo

Release-Data-Exporter (3 mins)

- Credentials for europepmc,ucsc, and ncbi
- This can be checked by looking at the log file in s3. It does not write that it completed to
- Look for WARN or ERROR in log (eg. https://reactome.s3.amazonaws.com/private/releases/89/data-exporter/logs/Main-06-17-2024_16.01.25.log.gz)
 - Lines that look like this are fine: "2025-04-02 17:54:29.640 [main] ERROR org.reactome.release.dataexport.datastructures.NCBIEEntry.getUniProtToNCBIGeneEntries(NCBIEEntry.java:85) - UniProt Accession Q92673-PRO_0000033164 (dbld 11059536) contains PRO identifier"
- standard out (jenkins console)

MSigDB-GSEA (1.5 mins)

- Email Project Manager the Reactome_GeneSet_XX.txt file that is found in s3 here: e.g. https://reactome.s3.amazonaws.com/private/releases/90/msigdb_gsea/data/Reactome_GeneSet_90.txt.gz
- This file gets uploaded to S3 as a part of the "Archive Outputs" step, so don't bother searching for it on the Release server.
- Email this

GOCAMS

- Message Dustin Ebert at GO that the GO CAMS are on production

Cleanup Snapshots

- Remove pre release release server snapshot

Create logs for Henning

See [here](#) for more details on ELK Maintenance

There is a spreadsheet that EBI uses to report on Reactome:

<https://docs.google.com/spreadsheets/d/13uVi-mAIR0iWY3gStvC6KsiV4LA5vP2P7zdxY3hM2Yg/edit#gid=1587583359>

This is populated by the script in /opt/gitroot/elk-setup/reporting/

Run:

```
$ run_all_reports.sh 20210501 20210601
```

Editorial Post-release

- See [Post-release steps in the Editorial SOP](#) (*Appendix R1*)

Appendices

Appendix R1: [Editorial Release SOP](#)

Appendix R2: [Pre-slice checklist](#)

Appendix R3: [Release database testing](#)

Appendix R4: [Cumulative statistics generation](#)

Appendix R5: [DOI submission protocol](#)

Appendix R6: [All release data quality assurance checks](#)

Glossary

Analysis core:

Build:

current (database):

Downloads folder:

gk_central (database): The name of the main curator database, found at <http://curator.reactome.org>.

Graph database:

graph-qa:

Job:

Module:

Ortholog:

Reactome (database):

Release_current (database):

Release_previous (database):

Release_Template

Slice: When the curator database (*gk_central*) is converted to the release database, it is said to be a 'slice' database. This database contains all curations that are considered ready for release, while excluding all work-in-progress curations. Typically, the name of the database is 'slice_test'.

Slice_current (database):

Slice_previous (database):

Slice_test (database): The name of the slice database that is generated for release from *gk_central*.

Stable_identifiers (database):

Stage:

Step:

Sections to be Written

- Email
- Updating Jenkins (or implement CRON job)
- S3
- Branches/Github access
- Plugins
- Shared library
- Release_Template
- In-process script approval
- Checking environmental variables
- Master configuration file
- Running a step
- Views within step pages
- SSH-CuratorTool connections
- Change graph.db permissions
- P2P
- Server storage management
- Neo4j in Docker
- Locations of everything on Release (jenkins folder, neo4j, solr, etc.)
- User input
- Hot-fixing config files and changes by creating tmp branch
- Troubleshooting graph-db/analysis core problems
- Configuring stepsToRun for DownloadDirectory
- Personal access tokens

V90_Sept 2024

- General update and reorganization
- Addition of appendices

V91_Dec 2024

- Moved some QA checks to Appendix R2
- Added step to update Topic.txt list if needed before final slice
- Added a check for literatureReferences with retraction_status containing Retracted or "no_use"

V92_April 2025

- Minor updates to Appendix R1, Editorial Release SOP and Appendix R3, Database testing

Appendix R1:Editorial release SOP

Overview:

- [Preslice checklist](#) (Appendix R2)
- [Preparing slice](#) (in main release SOP)
- [Slice QA](#) (in main release SOP)
- [Release document preparation](#)
- [Release database testing](#) (Appendix R3)
- [Post-release steps](#)

- [Release document preparation \(Appendix R7\):](#)
 - I. [Release date and version in title banner](#)
 - II. [Cumulative statistics for news](#)
 - III. [Disease variant statistics](#)
 - IV. [Computationally inferred](#)
 - V. [Front page News](#)
 - VI. [Editorial Calendar](#)

You will need to download any EHLID or other image being used for the News item locally. All files should be PNGs and have a width of 650 pixels (the width can and should be set from within Joomla).

- ~~I. Update Updating Front page Date and version~~

The banner information for that section of the front page needs to be edited manually in Joomla

- Log in to Joomla at release.reactome.org/staff.
- Go to home page.
- Click edit button next to module with stats in it
- Scroll to bottom to see Joomla interface
- Update release date and version in title banner
- Click save and close

- ~~II. Update Cumulative Stats for news.~~

Instructions for generating the cumulative statistic in [this document](#) (*Appendix R4*) are shown at the bottom of that page. In summary, human pathways, reactions, proteins, small molecules, total drugs, and literature references are derived from the Reactome [Front Page statistics](#); total protein, net protein, variants, and complexes are derived from the [Reactome statistics page](#); disease proteins, disease variants, and unique disease variants are derived from the [variant statistics script output](#) (https://release.reactome.org/download/current/disease_variant_ewas_mapping.tsv); and the remaining statistics (new proteins, all proteins forms, disease reactions, disease pathways, chemical drugs, and protein drugs) are generated through the database queries.

- ~~III. Disease variant statistics:~~

Disease protein and variant numbers needed for the release news are obtained from this file https://release.reactome.org/download/current/disease_variant_ewas_mapping.tsv which contains information about all Reactome genetic variants. It is created each release. This file is converted to a googledoc and placed in the appropriate folder in the [Release document preparation](#) directory. Two sheets need to be created: the UniqueDiseaseGenes sheet and the UniqueVariants sheet. The UniqueDiseaseGenes sheet is generated by copy-pasting column A (without the header) and removing the duplicates – **this gives a total number of unique disease genes**. The UniqueVariants sheet is generated by copy-pasting columns A (gene) and F (genetically modified residue) and removing duplicates – **this gives a total number of unique variants** (unique combination of gene and genetically modified residue irrespective of PTMs and localization).

- **Comparing disease stats between releases to identify new/lost content.**

Run this script to create the UniqueDiseaseGene and UniqueVariants sheet.

To this sheet add the VXX-1 (previous release version) UniqueDiseaseGene and UniqueVariants sheets

Run [this script](#) to to compare current and previous UniqueDiseaseGene and UniqueVariants.

- ~~IV. Updating Computationally Inferred Events~~

This step has been automated as part of the release pipeline but the text on the [Electronic Inference](#) Joomla page should be modified to reflect the updated statistics for orthology inference. Specifically, the numbers for organisms with the lowest and highest level of conservation must be updated according to the stats in the orthology report found at:

```
/usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update/report_ortho_inference_test_reactome_<release#>_sorted.txt
```

- ~~V. Update Front Page News~~

Collecting information needed for the news (and DOI submissions):

1. Pathways needing DOIs: Any pathway with a new external reviewer or author will need a DOI. Sort the reviewer management sheet to collect all reviewers for a given release (sort by release, then by projectID then status)
2. Copy all those set to ReadyForRelease to a new sheet and format this to contain all the columns needed for the [Team Author-Reviewer-DOI info for release documents](#) googledoc.
3. Add lines for any contributors that were authors (check project calendar entries).
4. Compare this list to the output of the DOI suggerter script (in releaseQA folder) to find any contributors missed in the above 2 steps. This step is necessary since pathways with just a few reactions modified by an external contributor might not get listed manually in the reviewer management sheet. Note that not all pathways suggested for a DOI should get one. If, for example, a revised reaction is used in multiple pathways, but the reviewer only looked at it in the context of one of them, the DOI should go on the pathway(s) that the reviewer checked. Note also that only the top most pathway checked by the contributor should get a DOI.
5. Report any discrepancies revealed by the DOI suggerter to curators.
6. Once any discrepancies have been resolved, carefully verify that all contributors associated with a “needs DOI” pathways are entered into the [Team Author-Reviewer-DOI info for release documents](#).
7. Curators should add any missing contributors identified by the DOI suggerter to the people sheet, and add the corresponding pathway to the project sheet and reviewer management sheets.

Formatting news item for the release

- ~~Create a googledoc called VXX news and place in [Version XX folder on Google drive](#).~~

It's helpful to use the previous release news version as a template for the static sections and delete the old content.

The Reactome [news](#) contains:

1. New and Updated pathways (hyperlinked to website)
2. New or updated Illustrations
3. New contributor names (hyperlinked to URL of choice)
4. Revised annotation statistics
5. Static content about Reactome Tools, Documentation, and Social Media links

- ~~1. Creating the new content section of the news item.~~

The information needed 1-3 can be retrieved from this [Team Author-Reviewer-DOI info for release documents](#) googledoc. Note that there are formulas that are used in these sheets to autopopulate the hyperlinked pathways and authors names, and testing site URL. These can be copied from the previous releases' sheet in the googledoc.

Pathway names should be hyperlinked to their pathways on release. URLs will have to be edited later in Joomla as described below.

Contributors' names should be linked to the URL of their choice (ORCID record by default).

- ~~2. Create Statistics paragraph~~

Enter data from the Reactome [Front Page Statistics](#), [Cumulative Stats accession table](#) (Appendix R4), the [Reactome Statistics Page](#), [release_stats](#) report (https://download.reactome.org/xx/stats/ordered_release_stats.html where xx is current release #), and the [variant information script report](#) into the paragraph below.

Annotation Statistics. Reactome comprises **XXXXX human reactions** organized into **XXXX pathways** involving **XXXXX proteins** and modified forms of proteins encoded by **XXXX different human genes**, **XXXXX complexes**, **XXXXX small molecules**, and **XXXX drugs**. These annotations are supported by **XXXXX literature references**. We have projected these reactions onto **XXXXX orthologous proteins**, creating **XXXXX orthologous pathways** in 14 non-human species. Version XX has annotations for **XXXX protein variants** (variant proteins) and their post-translationally modified forms in different localizations (**XXXX unique variants**), derived from **XXXX proteins**, which have contributed to the annotation of **XXXX disease-specific reactions** and **XXX pathways**.

***Note that the variant proteins and disease protein numbers are generated as described above*.**

- ~~3. Ask the outreach manager for any additional news items.~~

- **~~!! To be added to V93 News — draft announcement !!~~**

"As of Reactome V93, our Docker images are no longer available on DockerHub and have been migrated to AWS Elastic Container Registry (ECR). Users should pull images from AWS ECR moving forward. Updated access instructions are available in our documentation."

- **4. Adding News items to Joomla**

Two ways: through release.reactome.org/staff, or through back end release.reactome.org/administrator

Through release.reactome.org/staff (preferred method)

1. Login in to release.reactome.org/staff using personal credentials
2. Under “account”, click “publish an article”
3. Make sure you are in editor mode (top right corner of window)
4. add title to reflect current release version (VXX Released)
5. **IMPORTANT** - under Publishing tab, add
 - a. “news” to the category toggle. This allows you to find article again (under “About/News” drop down”) - otherwise it might get lost in dark back alleys of Joomla.
 - b. “release” to the tag toggle
6. To add an image (NOTE -see below. It is better to add image AFTER!! correcting the links to avoid needless scrolling through the image):
 - a. Save EHLD illustration as a .png
 - b. Upload image into Joomla by clicking on blue “image” button at bottom of page. Specify width 650 height 401
 - c. Click on /news folder as place to upload
 - d. Scroll down with the outermost scroll bar on right to “Upload file”, select local file, hit “start upload”
 - e. Click ‘Preview’ then Save
7. To update news items
 - a. Re-enter content tab edit mode (click wheel)
 - b. In “Editor” mode, paste news item from Google doc
 - c. Hyperlinks to pathway names and EHLD in news item will automatically be updated by adding “PathwayBrowser/#/R-HSA-” in front of #db_id so that links resolve to the right environment (release vs live site). Links to the tools in the news will also be automatically updated to the format “PathwayBrowser/#TOOL=AT”.
 - d. **In Publishing tab, verify Category “news” and Tags “release”.**
 - e. Preview and save
 - f. Check this site to make sure it is showing the correct release:
 - i. <https://release.reactome.org/tag/release>

Through release.reactome.org/administrator (less preferred method)

1. Login to release.reactome.org/administrator
2. Click “new article” in left menu
3. Add title (“VXX is released”) in top bar
4. Add image by clicking on image icon in formatting panel, switching to “news” folder and either selecting or uploading a png image
 - a. Dimensions should be width 650 height 401 style display:block margin-left:auto margin-right: auto

- b. Use formatting panel to center image
 - c. Insert a title above image, and hyperlink to pathway
5. Copy and paste news item from Google doc
6. On right hand side, default status should be “published”; set category to ‘news’
7. On Publishing tab (below “title” bar, next to ‘content’, ‘images and links’ etc - may need to enter a ‘start publishing’ date, if publishing shouldn’t happen immediately
8. Click save

Adding a new news item to live site

This is done automatically during pass to production.

ONLY follow this procedure below if you need to make a change after release (and ONLY after verifying that it's ok to do this with developers).

Login to release.reactome.org/staff, using own login, not superuser

Under “account”, select “Publish article”

Make article as [described above](#).

Confirm that it looks ok on release.

Once happy, Under account, select synchronization tool- this option is only available to certain people who have the appropriate permissions in Joomla - as of 2022, Lisa, Karen, Chuqiao (others)

Under “environment”, click development, to stage new article and confirm everything looks right.

i This tool synchronizes **Joomla Content** to the specified environment

Environment:

Development

*Be careful when selecting production.

Release Server Credentials

Your OS user:

Enter the user

Your OS password:

Enter the password

Key Phrase:

Enter the key phrase

Database

DB User:

Enter the database user

DB Password:

Enter the database password

Run

Enter appropriate passwords, hit run. Check logs to make sure no errors, then check that everything looks right on dev.reactome.org

If all ok, repeat same process after selecting “production” under environment. Check logs, then confirm all good on live site.

- ~~VI. Updating sandbox Editorial Calendar~~

A working document containing contributor and module information for each release (including, contributor name, email, module, module URL, DOI, etc) is kept [here](#). Before changing the live [editorial calendar](#), copy the most recent release calendar to the [release calendar archive](#) .

1. Using the Internal project Calendar, edit the *[sandbox calendar](#) to reflect the topics for the current release and the next release. This will be checked by release database testers.

- ~~VII. Updating the Editorial calendar~~

1. Once the release goes live, use the [sandbox calendar](#) content to edit the live [editorial calendar](#) to properly reflect the content for forthcoming and future releases.

*Since the external editorial calendar is always accessible to the public, this internal template is created to facilitate seamless updating of the public calendar page.

- ~~Release Database Testing~~

- ~~Prepare testing documents~~

The testing protocol can be found [here](#) in *Appendix R3*.

- ~~When testing is ready to begin, create a new testing report document in [this directory](#), using the previous report as a template (this will retain any “known issues” in the comments”. List the two curators testing at the top of the doc and have them use different colored text (blue or purple) for comments so developers know who to contact with any questions.~~
- ~~Create a new VXX tab in the [external link testing document](#) as described [here](#).~~
- ~~Send these instructions to the testers:~~

When Joel gives the ok to test the release database, here are the links you need:

1. Add your testing environment and testing status [here](#).
2. The testing instructions are [here](#).
3. The document with the [list of links to test](#) (also found in the testing report) is here. **Only one person needs to test the link document.** Be sure you're in the correct release tab.
4. The testing report is [here](#). (Note the color associated with your name and use that color text to enter issues so Joel/developers know who to contact with questions).

I find it easiest to work from the testing report as a checklist and use the links to the relevant sections in the testing instruction found in the testing report.

Be sure to create the testing links document before testing begins.

The editorial release manager (Lisa) will coordinate with testers (listed in this [google doc](#)) and set up a testing report release XX testing report (in this [Team drive directory](#))
She will alert the development release manager (Adam) when testing is complete, and the development release manager will work with other developers to decide on an official release2production date which will then be communicated to the group by email.

● Post-release steps

- 1. Make sure release date is correct on front page
- 2. ~~Update live editorial calendar [as described above](#)~~
- 3. ~~Updating Release Versions Calendar~~
 - ~~Update the [publicly facing release date table](#).~~
- 4. ~~Website checks~~
 - ~~Check that the new EHLs have been updated on the [live site](#).~~
 - 4a. ~~Verify updated live [editorial calendar](#) is visible.~~
 - 4b. ~~Check that the <https://reactome.org/tag/release> page is visible.~~
 - 4c. ~~Verify that the ORCID claiming feature works as described.~~
~~<https://reactome.org/userguide/claim-your-work#guide>~~
 - 4d. ~~Check the curator tool [download](#) on the download page.~~
 - 4f. Update the [pathways for external review](#)
- 5. Note any updates to the release/curation documents (PM to do)
 - See [here](#) for instructions
- 6. Updates resource files for the next release cycle
 - Change qa-properties to reflect the cutOffDate of the final slice
 - Change slicingTool.prop

On release

- 7. Update (using final slice date)
 - `/usr/local/reactomes/Reactome/production/Release/scripts/release-qa/release-qa/src/main/resources/qa-properties`
- 8. Update skip lists
 - Update skip lists based on previous release data in the `.../CuratorQA/resources/` and `.../ReleaseQA/resources/` directories

- 9. QA checks
 - Check for new events that are doRelease != true but have kept releaseDate entry
 - Run scripts to identify proteins in gk_central that have not been released:
 - Run script to identify events and entities that were previously released but now missing from live site: The script is on reactomerelease under the /usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update and can be run by typing 'perl unreleased_instances_with_released_stable_id.pl'. You can add '-h or --help' to see how to change the default options if you need to do that. By default, the script assumes the curation database is gk_central and the release database is the most current test_reactome_XX.
 - unreleased_EWAS.pl The script looks for EWASs that have been created but never released. It is on reactomerelease under the /usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update and can be run by typing 'perl unreleased_EWAS.pl'

- 10. Database cleanup
 - Drop all test databases from curator.

- 11. DOI submission

The protocol for [DOI submission](#) can be found here in *Appendix R5*.

V91_Dec 2024

Added notes on where to find release_stats report.

Added notes on how/where to document changes to release and curation documents.

V92_Mar 2025

Added precision on news preparation.

V93_June 2025

Cleaned up text, revised joomla update protocol for news item.

Revised release testing instructions

Appendix R2: Pre-slice checklist

Timeline:

- 1. No later than 6 weeks before the release QA start date,
 - Send out external review requests.
- 2. Six weeks before release QA starts
 - Curators send sketches of any necessary EHLd revision plans to the Illustrator
- 3. Three weeks before the final slice
 - Send out reminder letters to Reviewers
- 4. Three weeks before QA:
 - Ask curators to make sure reviewed pathways do `_release true`.
 - Review the weekly QA report for changes to EHLd pathways.
 - Verify that the curators have discussed these changes and that they have been entered into Cris' EHLd/Illustration tracking document.
- 5. Monday before database updates:
 - Remind curators to commit most recent diagrams, complete QA, set release dates and `dr_flags`.
 - Remind curators to confirm they have reviewed illustrations calendar in the Illustration management document and confirmed the status of planned EHLd changes with Cris. No changes should be made to this list after release QA starts.
 - Remind curators to confirm they have Reviewed/revised summations for pathways that are getting a new EHLds. Post the new EHLd / illustration list [here](#) under the appropriate release version tab, and send a reminder message to curators asking them to review the list.
 - Compare Topic.txt file (in `/usr/local/reactomes/Reactome/production/Release/scripts/release/website_files_update` on release) to the frontpage item list. Make sure any new topics are added to Topic.txt and submit any changes (new or removed topics) to Developers to adjust fireworks layout.
 - `Events _doRelease=true` and with `stableID released !=true` but `releaseDate` not current `releasedate`.
 - Events with `releaseDate` and `_doRelease = false`
- 6. The day before the Database updates,
 - send an email reminder to commit all local projects and that `gk_central` is closing at 9 AM.
- 7. The day of database updates:
 - Email that `gk_central` is closed and copy developers.
- 8. The day before release QA:
 - Run new `stableIdentifier` check in curator tool!
 - Update project calendar and reviewer management, and illustration management sheets to reflect final modules for that release.
 - Verify "need DOI" flags are set on pathways needing new/revised DOIs
 - Verify that all pathways with a new EHLd get a `hasEHLd true` attribute added.
 - Add the figure ULRs for pathways getting new figures/EHLd.

Format should be: `/figures/ehld/R-HSA-9679506.svg` (for pathways getting an EHLd)

or /figures/static/R-HSA-389356.svg (for pathways getting a static illustration).

- . Before the first run of release QA, verify with Cris that new/updated SVG file for EHLD and static images have been committed to cvs/github (Cris will send email when EHLD SVG file is available).
- Verify with Cris that .png files for new/updated EHLD are available on curator for RTF download files.
- . Update skiplists containing the ids of instances that will be skipped in the slice QA. Skip lists consist of the instances that should be skipped for a given check.
- Run CT diagram QA on gk_central to look for any missing objects in diagram, or missing reaction in diagram and run new stableID check.

V90_Sept 2024

V91 2024

Added a few QA checks that were previously part of the [release QA section](#) in the [Reactome release SOP](#).

Moved several checks that should be done after first slice.

Appendix R3: Release Database Testing

*****CLEAR YOUR BROWSER CACHE BEFORE STARTING TO TEST*****

Setting up to test

Log the OS, browser, and browser version used in this [googledoc](#). The following tests are primarily designed to test the Reactome website's functionality. Checking the actual data is not intended to be part of this, though if you do notice something, please report it to the internal list.

It would be very helpful if testers could coordinate amongst themselves to try as wide a range of browsers and operating systems as possible.

Put together a checklist of things that need fixing and post it to the internal list. If everything ran without problems, send a note to the internal list anyway, just so that we know you have completed testing! Please include the OS, browser and browser version when reporting results.

Reporting a problem or bug

For tracking purposes, please use the release XX testing report (in this [Team drive directory](#)) to list all issues encountered while testing. This way, the other tester(s) and the developers can regularly refer to this and see/comment on the issues and their resolution status. The two testers should be listed at the top of the doc and use different text color so the developers know who to contact for any questions on issues logged.

Testing

Go to <https://release.reactome.org/>.

Front Page

- The front page should look like this:

Find Reactions, Proteins and Pathways

[Go!](#)


Pathway Browser

Visualize and interact with Reactome biological pathways



Analysis Tools

Merges pathway identifier mapping, over-representation, and expression analysis



ReactomeFIViz

Designed to find pathways and network patterns related to cancer and other types of diseases



Documentation

Information to browse the database and use its principal tools for data analysis



Reactome Research Spotlight

[September 1, 2024] In the August 2024 issue of *Nature Microbiology*, [Bracha et al.](#) use Reactome expression analysis to confirm that they successfully delivered multiple large (>100 kDa) therapeutic proteins across the blood-brain barrier into target neurons in mice using engineered *Toxoplasma gondii* secretion systems.

[ARCHIVE](#)


Why Reactome

Reactome is a free, open-source, curated and peer-reviewed pathway database. Our goal is to provide intuitive bioinformatics tools for the visualization, interpretation and analysis of pathway knowledge to support basic research, genome analysis, modeling, systems biology and education.



The development of Reactome is supported by grants from the US National Institutes of Health (U24 HG012198) and the European Molecular Biology Laboratory.



Latest News

V89 Released
New Publication in Database (Oxford)
V88 Released
V87 released
V86 Released
V85 released
V84 released
Reactome is hiring!



Version 89 released on June 16th, 2024



2,711

Human Pathways



15,326

Reactions



11,495

Proteins



2,127

Small Molecules



1,047

Drugs



38,895

Literature References



Do you need help ?



Use Reactome!



For Developers



Citing us



Contact Us



API and Data access

Reactome Frontpage

- Verify that correct version number and release date is listed at the bottom of the homepage.
- Test that all buttons link out to the expected places and that the links to each participating institute (at the bottom of the page) work.
- Make sure that the news is up-to-date. Please check the news item and links.
- Does the search work? Enter 'p53'; you should get over 1700 results. Verify that search results are confined to Homo sapiens (or entities without species).
- Also, test :
 - A newly added pathway, reaction, and complex (see [Project calendar](#) for new pathway).
 - An old pathway, reaction, and complex.

Front page navigation bar

- In the navigation bar, Scroll over these items in the Navigation Bar: 'About', 'Content', "Docs", 'Tools', 'Community', and 'Downloads'. They should all have a drop-down menu.
- Mouse over each of these drop-down menus in turn. Ensure that each link in each drop-down menu leads to a valid web page.*Note that editorial and release calendars are not updated until after release.
- Under the "About" -> "News" Test all the links in the most recent News Item.
- Under the "About" -> "Statistics" current release and date listed
- Under the "Content" -> "Table of Contents", the "NEW" and "UPDATED" flags are correct (check one new/updated pathway from the [Project calendar](#)).
- From the "Content" menu, open "DOI", search current release date, and verify that "NEW" and "UPDATED" flags appear on the correct Pathways (check one of each from [this list](#)).
- Check the sandbox for the live editorial calendar [here](#). Note that the tab pointing to the actual calendar will not display the updated calendar until the release has gone live.
- From the "Docs" menu, check that the "Computational Inferred Events" page displays the correct image for the current release.
- Click on the "Download" menu item. Verify that **all** of the links on the page are valid and that the files are generated. Check that all the files can be downloaded (except for the curator tool, which cannot be downloaded from release.reactome.org). #Note" that testers should verify that BioPax2/3 zip folders contain OWL files for other species in addition to Homo Sapiens.

Pathway Browser

Basic Navigation and search

Go to <http://release.reactome.org/PathwayBrowser/>

Navigation

- The Pathway Browser should appear, with a list of top-level pathways on the left
- Verify that the navigation icons to the right of the species selection dropdown (citation, analysis, tour, and layout) and those at the top left of the fireworks panel ("search," "Show all," "Open pathway diagram" (when pathway selected), and "Open Voronoi visualization") work and that new pathways are visible in the Voronoi view. For a list of the new pathways, see the [project calendar](#) for that release).
- Verify that only these 16 species are shown in species dropdown list: *Rattus norvegicus*, *Gallus gallus*, *Drosophila melanogaster*, *Caenorhabditis elegans*, *Bos taurus*, *Saccharomyces cerevisiae*, *Schizosaccharomyces pombe*, *Dictyostelium discoideum*, *Plasmodium falciparum*, *Sus scrofa*, *Mycobacterium tuberculosis*, *Canis familiaris*, *Xenopus tropicalis*, *Danio rerio*, *Mus musculus*, and *Homo sapiens*.
- Click on each of the top-level pathway names in the pathway hierarchy; a diagram should be drawn on the right for each of them.
- Change the species to 'Mus musculus'.
- Once again, click on each of the pathway names on the left-hand side in turn and verify that a diagram is drawn on the right for all of them. Be sure to check the display of the 'Mus' (or other projected species) pathway one old and one "new" pathway.
- Change the species back to 'Homo sapiens'.
- Click on "[DNA repair](#)"; an EHL diagram should appear. Unfurl the pathway.
- Double-click to enter "[DNA damage bypass](#)". This should take you to a pathway showing entities. Scrolling over the two subpathways in the hierarchy, you should see the contained reactions light up in yellow in the diagram. Unfurl one of the subpathways and select a reaction. This should be highlighted in the ELV diagram.
- Use the scroll wheel of your mouse; the diagram should zoom in and out.

Diagram features

- Check the functionality of the other in-diagram options in the settings panel:
 - Diagram key (on right side of diagram panel) is visible
 - Download diagram
 - Export diagram and export content (check a formats)
 - Test interaction overlay options (test overlay and downloads)
 - Zoom in, out, and center

Search

- Search on USP10 in the [DNA damage bypass](#) diagram.
- Check that, in the diagram USP10 (and any complex or reactions containing it) are highlighted in yellow when you scroll over its name and highlighted in blue when you click on the name.
- Click the "All Diagrams" tab in the In-Diagram search to view search results across all diagrams. The initial hits and the number of hits in the In-Diagram search window should differ between USP10 in Diagram vs USP10 in All Diagrams.

Details Panel

Pathway

Go to [DNA damage bypass](#) pathway.

In the details panel below the diagram you should see the details section with 6 tabs. Check that the information in all tabs is showing up (if applicable)

1. Description

- a. You should see the name of the pathway, StableID, contributors, descriptive text, contained events, Computationally inferred events (check one), references, and a diagram.

2. Molecules

- a. Open the "Molecules" tab. Select one of the uniprotIDs under "Proteins". The number next to the protein represents the number of molecules in the diagram that contain it. Clicking on the icon (green circle) will cycle through all instances of that protein in the diagram. Clicking on a protein/complex/set in a diagram should shade all of the contained entities in the molecule list.
- b. Test filters and downloads.

3. Structures

- a. Verify structures are shown for proteins.

4. Analysis

5. Expression

6. Downloads

- a. Test all pathway and pathway diagram download formats

Reaction

- Go to the reaction ["Caspase-8 processing within TLR4 complex"](#)
- Check for name, StableID, Summation, Computationally inferred events (check one), Input, output, Catalyst, References, Authored, Reviewed.
- Search for the DISC:procaspase-8-dimer icon in the ELV; confirm that the text in the details panel changes to describe it, and
- In the "Description" tab, select the complex icon, unfurl complex to see components. Unfurl all the way to the referenceEntity.

Context Sensitive Menus

- Go to ["Circadian Clock"](#)
- Search on "SIRT1"; Right click on entity icon in the diagram you should see a menu : Molecule, Pathways, Interactors) Click on " Pathways"
- Mouse over it; you should see a listing of all* of the pathways in which it is found.
- Click on one; it should take you to a different pathway diagram;
- On SIRT1 in diagram,click on the red circle to display the interactors and reclick to hide them.

- Adjust the confidence level in the sliding scale. The number of interactors shown should decrease as the confidence score increases.
 - Verify that the diagram settings panel (on the right side of the diagram panel) Color profiles, Interactors, and information features work properly.
 - Test interactor download feature on right of the sliding scale (confidence) bar.
-
- In the details panel at the bottom of the page, you should see the details for the selected entity.

Analysis Tools

- ****Note**** From the front page when clicking the pull-down Tools menu, only the "analyze data" option opens the analysis page with the correct tab open (Analyse gene list). From the same pull-down menu, choosing the other 3 analysis options opens the analysis page but always at the "Analyse gene list" tab open.
- ****Note**** For "Analyse Gene Expression" the "B cell RNAseq example" is only compatible with the ssGSEA analysis. For PADOG and Camera methods, the return message is "Oops! An error has occurred. Analysis failed, please contact help@reactome.org. Failed to convert dataset 'B cell scRNAseq example'. Insufficient samples in group 'Plasma.like'. Each group must at least contain 3 samples for accurate results."

Analysis tools

Analyse gene list

Analyse gene expression

Species Comparison

Tissue Distribution

Your data **Options** **Analysis**

Step 1: Select a file from your computer or paste your own data and click on the corresponding "Continue" button.

Select data file for analysis: No file chosen

Paste your data to analyse or try example data sets:

Paste your data here or select an example from the right >>

Some examples:

- UniProt accession list
- Gene name list
- Gene NCBI / Entrez list
- Small molecules (ChEBI)
- Small molecules (KEGG)
- Microarray data
- Metabolomics data
- Cancer Gene Census (COSMIC)
- Tissue Specific Expression (HPA)

Analyse gene list

- 1) Click "Analyze Data" button on the [front page](#)
2. Test all available sample data sets following steps 3-10.
- 3) In the Analysis details panel, Click "Results". See list of pathway names, and a series of numerical values representing entities and reactions, ratios, statistics, and species.
- 4) Select a pathway name listed in Analysis details tab. Make sure the pathway diagram loads, and the pathway participants in diagram will be colored if hit.
- 5) Select a complex that is a hit. Right-click and opt to "Display Participating Molecules". A list of components with color indicating if component was hit should appear.
- 6) Optional: Select a set that is a hit. Right-click and opt to "Display Participating Molecules". A of the contained entities colored to indicate if they were hit should appear.
- 7) Apply filters (funnel shaped icon at the upper right corner of Analysis details panel) and look for results changing.
- 8) **Important:** *At the top of the results page menu bar, you can restrict results by resources by selecting the "Results for:". Depending on the data set, change the identifier type from TOTAL to CHEBI, UNIPROT, COMPOUND, DRUG The table of results will change to reflect the selected identifier type. Be sure that results of all classes are shown and downloadable. This "results" item is hidden under the funnel-shaped icon at the extreme right of this line in the results output -*



- 9) In the Analysis details panel, Click "Identifiers not found". A list of Entities not found in the data set should appear.
- 10) In the Analysis details panel, Click "Downloads" and verify all files are created.
- 11) Try to upload [GBM data set](#) to test

Species comparison

Click "Analyze Data" button on the [front page](#) and go to Species Comparison

- Test one species.
- Click "Analyze Data" button on the front page (also verify analysis icon on top menu bar of Pathway Browser page opens the analysis window in the central panel of the page).
- On the left side of the page, click the Species Comparison button; select species "Mus musculus" from 'Compare Homo sapiens with:' drop-down menu.
- Click the 'Go' button.
- Check analysis results in the Analysis details tab. See list of pathway names, and a series of numerical values representing entities and reactions, ratios, statistics, and species.


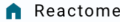
- Select "Metabolism" listed in the Analysis details tab. Make sure the pathway diagram loads, and the pathway participants in the diagram are coloured if hit.
- Select "Metabolism of carbohydrates".
- Select a complex that is a hit. Right-click and opt to "Display Participating Molecules". A list of components with color indicating if a component is an ortholog should appear.
- Optional: Select a set that is a hit. Right-click and opt to "Display Participating Molecules". A list of components with color indicating if a member is an ortholog should appear.
- Check that the number of molecules matched/total number of molecules and FDR values are added to the right side of pathway names in the Event Hierarchy Panel.
- Within the Event hierarchy, click the pathway "[Circadian Clock](#)". There should be a different pathway diagram to the right of the pathway list.
- In the top left-hand corner of the diagram, use the zoom feature to zoom out. You should see that about two dozen entities colored yellow and a couple in blue.
- Apply filters (funnel shaped icon at the upper right corner of Analysis details panel) and look for results changing.
- In the Analysis details panel check the results and results downloads
- In the Analysis details panel, Click "Downloads check pathway format and pathway diagram downloads ": verify all files are created.

Tissue distribution

Click "Analyze Data" button on the [front page](#) and go to Tissue distribution and test several tissues.

- Zoom into a diagram and make sure that the dynamic display toggles between the tissues and changes are seen.
- Repeat steps 3-10 in Analyse gene list section above.

Analyse gene expression(GSA)



Welcome to Reactome Pathway Analysis



ReactomeGSA is a powerful tool for understanding the biological significance of gene sets and their expression. It allows researchers to uncover the functional relevance of a list of genes, associated with quantitative data, in the context of biological pathways and processes.



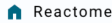
How to Get Started:





1. For your first time using ReactomeGSA, follow our guided tour.
2. Upload Your Gene Expression Dataset: Input a tabular data with gene or protein identifiers by row, and sample by column.
3. Run the Analysis: Start the analysis to identify enriched pathways and gain biological insights.
4. Explore Results: Visualize and explore the results to understand the functional context of your gene list.
5. Interpret Findings: Interpret the findings in the context of your research goals and biological questions.

Get started with ReactomeGSA analysis today and unlock the power of pathway enrichment analysis!

For detailed information, please see [\[Griss J et al, MCP, 2020\]](#).





Step 1: Select one of the available analysis methods

PADOG


Weighted gene set analysis method that down-weights genes that are present in many pathways.

Camera

A gene set analysis algorithm similar to the classical GSEA algorithm as implemented in the limma package.

ssGSEA

The ssGSEA approach to derive pathway expression values for every sample. Note: The Reactome visualization is only available for up to 15 samples.


Continue

Test dataset check

Perform analysis on test data using all analysis methods:

PADOG

CAMERA

ssGSEA (only one dataset)

Opt to:

Create REACTOME visualizations

Create reports

E-Mail

Public dataset checks:

Perform analysis using public datasets using one of the following algorithms:

PADOG

CAMERA

Fetch dataset via the panel *Public Datasets*, use the section *GEOQuery* and search for a microarray dataset using the geo accession number (e.g. GSE1563)

Choose statistical design according to the chosen dataset.

In the statistical design for GSE1563, select clinical status as "Comparison factor" and then control healthy blood donor as "1st group" and "PBL sample from transplant patient with kidney with no clinical evidence of rejection as "2nd group" without selecting any covariates. Save the dataset. Continue to GSA analysis.

Opt to:

Create REACTOME visualizations

Create reports

E-Mail

Visualization checks:

1) In the Analysis details panel, Click "Results". See list of pathway names, and a series of numerical values representing entities and reactions, ratios, statistics, and species.

2) Select a pathway name listed in Analysis details tab. Make sure the pathway diagram loads, and the pathway participants in diagram will be colored if hit. Verify that the display is cycling through the different samples data.

3) In the Analysis details panel, Click "Identifiers not found". A list of Entities not found in the data set should appear.

4) In the Analysis details panel, Click "Downloads" and verify all files are created.

Reports check

Check report was generated and looks complete and that results were emailed.

External link check

Test this [list](#) generated each release using this query (see the appropriate version tab in this sheet):

```
MATCH (rd:ReferenceDatabase)-[]-(cr) WITH rd, rd.displayName AS displayName, rd.accessUrl AS  
accessUrl, COLLECT(cr.identifier) AS identifiers RETURN displayName, replace(accessUrl,  
"###ID###", HEAD(identifiers)) as representativeIdentifier ORDER BY displayName;
```

***Note:** There is no reason for both database testers to do this check, so once a tester has done this, they should indicate that in the testing doc so the other tester doesn't bother.

New DOIs

Check [this sheet](#) (under the relevant tab for the current release) for a new pathway with a DOI. Look up the DOI in crossref to make sure it points to the pathway in Reactome.

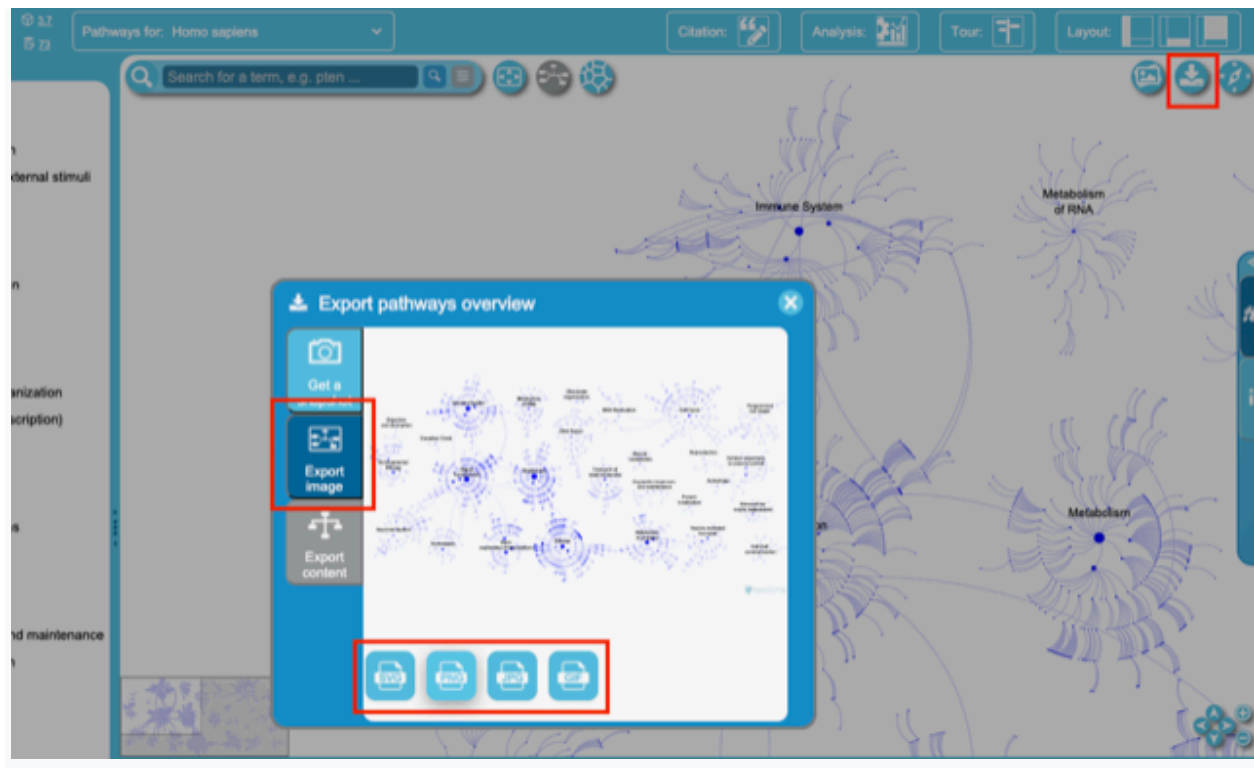
Test exporters

Fireworks exporter:

- Export fireworks view in different format svg, jpg.
- Run Overrepresentation Analysis and export fireworks image in different formats svg, jpg. (*note there is no color overlay in .ppt export)
- Test microarray data set example where there is a visible change between 18-24 hrs. Export both "time frames" in different formats (svg, jpg and gif) and make sure differences are detected.
- Run GSA Analysis and export different formats svg, jpg. Most important here is to check the GIF format.
 - Select one "column" for the expressed column and export different formats svg, jpg.

Diagram-exporter:

- Select one diagram (Preferably a new or updated) and export different format svg, jpg.
- Run Overrepresentation Analysis and export different formats svg, jpg. (*note there is no color overlay in .ppt export)
- Test microarray data set example for [this pathway](#) where there is a visible change between 18-24 hrs. Export both "time frames" in different formats (svg, jpg and gif) and make sure differences are detected.



Post-release checks

Editorial Release manager checks

- Check that the new EHLs have been updated for the [live site](#).
- Update live [editorial calendar](#) is visible.
- Check that the <https://reactome.org/tag/release> page is visible.
- Verify that the ORCID claiming feature works as described.
- <https://reactome.org/userguide/claim-your-work#guide>

V90_Sept 2024

- General update and restructure
- Add GEO query testing instruction

V91_Dec 2024

- Added a link to a new link testing reporting document

V92_Mar 2025

- Added a note to test expression tab
- Added a section under Release Database Testing describing document preparation (creation of the testing report and the link testing document).

V94_Sept 2025

- Changed the

V90_Sept 2024

- General update and restructure
- Add GEO query testing instruction

V91_Dec 2024

- Added a link to a new link testing reporting document

V92_Mar 2025

- Added a note to test expression tab
- Added a section under Release Database Testing describing document preparation (creation of the testing report and the link testing document).

V94_Sept 2025

- Changed the

Appendix R4: cumulative statistics generation

ver	date-cal	prot	netprot	isoforms	complexes	rxn*	litRef	chemicals	newprot	EWAS	DiseaseProt	DiseaseVar	diseaseRxn	diseasePat	chem drug	protein drug	total drug	pathways*
10	20040706	763				1247	894											
11	20041027	891				1512	1083		128									
12	20050201	946				1554	1161		55									
13	20050411	956				1586	1201		10									
14	20050613	1042				1711	1312		86									
15	20050926	1095				1524	1410		53									
16	20060123	1179				1639	1608		84									
17	20060424	1369				1828	1886		190									
18	20060802	1452				1784	2078		83									
19	20061116	1473				1845	2241		21									
20	20070228	1576				1947	2480		103									
21	20070515	1921				2113	2783		345									
22	20070904	2203				2241	3144		282									
23	20071211	2293				2327	3244		90									
24	20080311	2499				2345	3587		206									
25	20080630	2724				2567	3998		225									
26	20080930	2921				2871	4167	582	197									
27	20081217	3368				3071	4496	641	329									
28	20090401	3702				3203	4769	659	273									
29	20090624	4181				3335	5078	670	309									
30	20090930	4605				3541	5571	697	424									
31	20091215	4757				3669	5970	704	152									
32	20100316	4918				3543	6282	712	161									
33	20100615	5133				3658	6757		215									
34	20100921	5292				3837	7305	782	159									
35	20101214	5759				4166	8107	831	467									
36	20110315	5923				4247	8473	865	164						9	1	10	
37	20110614	6248				4354	8973		325						13	1	14	
38	20110920	6319				4518	9392	893	71						14	1	15	
39	20111206	6547				4827	10039	932	128						15	1	16	
40	20120313	6766				5038	10692	980	219						20	1	21	
41	20120612	7066				5568	11824	1100	300						26	1	27	
42	20120918	7161				5723	12334	1203	95						26	1	27	
43	20121204	7309	6572			6004	13060	1307	148						38	1	39	
44	20130312	6977	6745			6198	13588	1347	-332	13825					38	1	39	
45	20130611	7187	6984			6478	14556	1421	210	14348					38	1	39	
46	20130918	7293	7088			6744	15107	1421	106	14767	69	420			38	1	39	
47	20131204	7414	7203			6849	15555	1423	121	15065	69	420			38	1	39	
48	20140312	7545	7329			7075	16244	1443	126	15479	77	463			38	1	39	
49	20140610	7684	7465			7332	16900	1462	136	15965	82	476			39	1	40	
50	20140930	7822	7600			7462	17248	1419	135	16216	94	527			40	1	41	
51	20141211	7982	7760			7686	17939	1428	160	16822	137	732			40	1	41	
52	20150319	8183	7954			8169	18900	1449	194	17321	149	764			40	1	41	
53	20150617	8360	8232			8369	19562	1454	278	17761	168	850			41	1	42	
54	20150922	8701	8467			8770	20708	1540	235	18658	249	1155			41	1	42	
55	20151215	8930	8695			9052	21604	1618	228	19365	257	1209			41	1	42	
56	20160323	9483	9245			9584	22838	1665	550	20391	278	1263			44	1	45	
57	20160627	9868	9629			9865	23860	1679	384	21283	278	1264			44	1	45	
58	20160919	10461	10221			10168	24974	1710	592	23215	285	1331			44	1	45	
59	20161212	10624	10381			10391	25449	1735	160	23661	285	1333			44	1	45	
60	20170327	10907	10658			10754	26384	1763	277	24425	285	1334			44	1	45	
61	20170612	10940	10691			11042	26859	1762	33	24613	285	1334			44	1	45	
62	20170912	10973	10719			11302	27526	1768	28	24704	285	1334			44	1	45	
63	20171213	10996	10739			11426	27694	1764	20	24775	287	1334			49	1	50	
64	20180321	11030	10769			11574	28254	1867	30	24974	289	1339			49	4	53	
65	20180612	11032	10780			11896	28436	1880	11	25192	289	1339			60	4	64	
66	20180912	11049	10785			12047	28829	1948	5	25326	293	1391			125	4	129	
67	20181213	11066	10799			12778	29454	1827	14	25417	299	1409			179	4	183	
68	20190320	11100	10832			12416	29885	1854	33	25515	304	1416			182	5	187	
69	20190612	11109	10840			12505	30027	1857	8	25622	305	1475			195	5	200	
70	20190910	11138	10867			12608	30398	1856	27	25849	308	1599			217	5	222	
71	20191210	11152	10877			12770	30721	1863	10	26219	310	1816			219	6	225	
72	20200317	11193	10915			12986	31237	1865	38	26433	315	1890			230	7	237	
73	20200610	11208	10930			13248	32150	1869	0	27328	327	2620			347	22	369	
74	20200915	11207	10929			13416	32297	1854	0	27330	328	2624			367	27	394	
75	20201208	11215	10936			13534	32493	1854	7	27729	335	2933			367	27	394	
76	20210323	11362	11079			13732	33453	1856	143	28062	337	2949			386	29	415	
77	20210609	11374	11090			13827	33752	1857	11	29680	347	4464			402	30	432	
78	20210924	11011	10726			13890	34025	1940	-363	29466	352	4603			468	39	507	
79	20211208	11363	11047			13960	34252	1987	352	30313	352	4603			490	37	527	
80	20220407	11378	11366			14108	34703	1987	51	30485	352	4977			495	37	532	
81	20220609	11383	11094			14246	35629	1986	-1	30505	352	4977			941	81	1022	
82	20220914	11393	11103			14398	35965	1998	9	30548	352	4977			949	86	1035	2601
83	20221207	11442	11152			14471	36290	2002	49	30656	361	4977			950	86	1036	2610
84	20230317	11371	11080			14516	36444	2002	-72	30095	353	4983			950	86	1036	2615
85	20230607	11396	11103	204	14277	14628	36706	2004	23	30155	354	4919	1,659	707	951	86	1037	2629
86	20230901	11448	11154	208	14441	14803	37156	2025	51	30338	354	4919	1667	707	952	86	1038	2647
87	20231128	11392	11180	212	14596	15046	37933	2120	31	30451	357	4942	1796	723	956	90	1046	2673
88	20240315	11442	11226	216	14,789	15,212	38549	2128	46	30585	359	4944	1802	725	957	90	1047	2698

89	20240603	11495	11279	216	14,897	15,326	38,895	2,127	53	30733	376	4857	1802	725	957	90	1047	2,711
90	20240911	11506	11289	217	15,299	15492	39318	2127	10	30892	391	4943	1808	746	967	90	1057	2742
91	20241122	11541	11323	218	15,383	15,591	39806	2125	34	31055	391	4943	1808	746	967	90	1057	2751
92	20250314	11,574	11,356	218	15,486	15,672	40,321	2,130	33	31168	392	4947	1809	747	967	90	1057	2,769
93	20250611	11,615	11,396	218	15,621	15,890	41,048	2,172	40	31906	393	5510	1881	764	978	90	1,068	2,799
94	20250915	11,630	11,410	220	15,751	16,002	41,373	2,176	14	31991	392	5507	1888	763	975	95	1,070	2,825

[Get dates from Release Date History file.](#)

From V88, stats are collected from:

[Reactome Statistics Page](#) (automatically generated)

[Reactome Front Page Stats](#) (automated) see https://download.reactome.org/XX/stats/summary_stats.json

[Variant statistics script](#) report (as of V89)

[CT queries on slice](#) (see pre -V88 methods below)

[Stats page](#) [or](#) [Front page](#)

Prior to V88 statistics were generated as follows:

Get data from slice_test at host: 127.0.0.1, port: 1234, user: curator

Prot = ReferenceGeneProducts whose species = Homo sapiens

netprot = prot - referenceIsoforms whose species = Homo sap

complexes = complexes whose species = Homo sapiens

rxn = reactionlikeEvents whose species = Homo sapiens

litRef = literatureReferences

chemicals = referenceMolecules

chemicalDrugs - check out all chemicalDrug instances into a local project; the tally is the number of referenceTherapeutics l

proteinDrug-check out all ProteinDrug instances into a local project; the tally is the number of referenceTherapeutics brought along as referrers.

newprot = netprot(this release) - netprot(previous release)

EWAS = EWAS whose species = Homo sapiens

DiseaseProt and DiseaseVar: (prior to V88) calculated by checking out all geneticallyModifiedResidue instances. DiseaseProt is the number of human referenceGeneProducts brought along as referrers and DiseaseVar is the total number of geneticallyModifiedResidue instances mini

pathways = pathways whose species = Homo sapiens

DiseaseRxn=all human reactionlikeEvent disease not null

DiseasePath=all human pathways disease not null

V78 loss of "new proteins due to the HLA cleanup (link to doc?)

V84 loss of proteins due to the disease variant cleanup (link to doc?)

Corona stats - updated for version 76

10 proteins

10 net proteins

152 EWASs

124 reactions (relatedSpecies matches Human SARS coronavirus)

108 complexes (79 purely viral, 29 virus + host)

ver	complexes	complexes	rxn**	rxn change	litRef	litRef change	EWAS
79			13960		34252		30313
80			14108	148	34703	451	30485
81			14246	138	35629	926	30505
82			14398	152	35965	336	30548
83			14471	73	36290	325	30656
84			14516	45	36444	154	30095
85	14277		14628	112	36706	262	30155
86	14441	164	14803	175	37156	450	30338
87	14596	155	15046	243	37933	777	30451
88	14,789	193	15,212	166	38549	616	30585
89	14,897	108	15,326	114	38,895	346	30733
90	15,299	402	15492	166	39318	423	30892
91	15,383	84	15,591	99	39806	488	31055
92	15,486	103	15,672	81	40321	515	31168

[Get dates from Release Date History file](#)

From V88, stats are collected from:

[Reactome Statistics Page \(automatically generated\)](#)

[Reactome Front Page Stats \(automated\)](#) see https://download.reactome.org/XX/stats/summary_stats

[Variant statistics script report \(as of V88\)](#)

[CT queries on slice](#) (see pre -V88 methods below)

[Stats page](#) or [Front page](#)

Prior to V88 statistics were generated as follows:

Get data from slice_test at host: 127

Prot = ReferenceGeneProducts whose species = Homo sapiens

netprot = p

complexes = complexes whose species = Homo sapiens

rxn = reactionlikeEvents whose species = Homo sapiens

litRef = literatureReferences

chemicals = referenceMolecules

chemicalDrugs - check out all chemicalDrug instances into a local project; the tally is the newprot = netprot(this release) - netprot(previous release)

EWAS = EWAS whose species = Homo sapiens

DiseaseProt and DiseaseVar: calculated by checking out all geneticallyModifiedResidue instances. D

pathways = pathways whose species = Homo sapiens

DiseaseRxn=all human reactionlikeEvent disease not null

DiseasePath=all human pathways disease not null

V78 loss of "new proteins due to the HLA cleanup (link to doc?)

V84 loss of proteins due to the disease variant cleanup (link to doc?)

Corona stats - updated for version 76

10 proteins

10 net proteins

152 EWASs

124 reactions (relatedSpecies matches Human SARS coronavirus)

108 complexes (79 purely viral, 29 virus + host)

EWAS change	DiseaseVar	DiseaseVar change	total drug	total drug
	4603			
172	4977	374		
20	4977	0		
43	4977	0		
108	4977	0		
-561	4983	6		
60	4919	-64		
183	4919	0		
113	4942	23	1046	
134	4944	2	1047	1
148	4857	-87	1047	0
159	4943	86	1057	0
163	4943	0	1057	0
113	4947	4	1057	0

[ats.json](#)

DiseaseProt is the number of human referenceGeneProducts brought along as referrers and DiseaseVar is

; the total number of geneticallyModifiedResidue instances minus the number that refer to non-h

uman referenceGeneProducts

version	date	Hs proteins	EWAS	Disease Prot	Disease Var	chemicals	chem drug	protein drug	reactions	literature references
72	Mar-20	10915	26433	315	1890	1865	230	7	12986	31237
76	Mar-21	11079	28062	337	2949	1856	386	29	13732	33453
	72-76	164	1629	22	1059	11	156	22	746	2216
80	Mar-22	11098	30485	352	4977	1987	495	37	14108	34703
	76-80	19	2423	15	2028	131	109	8	676	1250

ver	date-cal	prot [*]	netprot	variants	complexes
87 Manual	20231128	11481*	11185		14596
87 Statistics Page		11392	11180		14596
87 front page		11392	N/A		N/A
88 Manual	20240315	11533	11232		14789
88 Statistics Page		11442	11,226		14,789
88 front page and Adam's script		11442	N/A		N/A
v89 adam's script					
V92 adam's script					
V92-V88 differences					

[Statistics Page \(automatically generated\)](#)

[Front Page Stats \(automated\)](#)

[Variant statistics script](#)

Manual

^{*} [Stats page](#) or [Front page](#)

CT queries on slice (see pre -V88 methods below)

indicates data used for that version news

Prior to V88 statistics were generated as follows:

Get data from slice_test at host: 127.0.0.1, port: 1234, user: curator

Prot = ReferenceGeneProducts whose species = Homo sapiens

netprot = prot - referenceIsoforms whose species = Homo sapiens

complexes = complexes whose species = Homo sapiens

rxn = reactionlikeEvents whose species = Homo sapiens

litRef = literatureReferences

chemicals = referenceMolecules

chemicalDrugs - check out all chemicalDrug instances into a local project; the tally is the number of

newprot = netprot(this release) - netprot(previous release)

EWAS = EWAS whose species = Homo sapiens

DiseaseProt and DiseaseVar: calculated by checking out all geneticallyModifiedResidue instances. |

pathways = pathways whose species = Homo sapiens

DiseaseRxn=all human reactionlikeEvent disease not null

DiseasePath=all human pathways disease not null

rxn*	litRef	chemicals	newprot	EWAS	DiseaseProt	DiseaseVar
15046	37954	2051	31	30451	357	4942
15046	N/A	N/A	N/A	N/A	N/A	N/A
15046	37933	2120	N/A	N/A	N/A	N/A
15212	38570	2056	47	30585	359	4944
15,212	N/A	N/A	N/A	N/A	N/A	N/A
15212	38549	2128	N/A	N/A	376	4857
					376	4857
					392	4947
					16	90
					16	90

referenceTherapeutics brought along as referrers.

DiseaseProt is the number of human referenceGeneProducts brought along as referrers and Disea

diseaseRxn	diseasePath	chem drug	protein drug	pathways [*]	total drug
1796	723	956	90	<u>2673</u>	N/A
N/A	N/A	N/A	N/A	2673	N/A
N/A	N/A	N/A	N/A	2673	1046
1802	725	957	90	2698	1047
N/A	N/A	N/A	N/A		
N/A	N/A	N/A	N/A	2698	1047

aseVar is the total number of geneticallyModifiedResidue instances minus the number that refer t

ver EWAS

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51

13825
14348
14767
15065
15479
15965
16216
16822

523
419
298
414
486
251
606

52	17321	499
53	17761	440
54	18658	897
55	19365	707
56	20391	1026
57	21283	892
58	23215	1932
59	23661	446
60	24425	764
61	24613	188
62	24704	91
63	24775	71
64	24974	199
65	25192	218
66	25326	134
67	25417	91
68	25515	98
69	25622	107
70	25849	227
71	26219	370
72	26433	214
73	27328	895
74	27330	2
75	27729	399
76	28062	333
77	29680	1618
78	29466	-214
79	30313	847
80	30485	172
81	30505	20
82	30548	43
83	30656	108
84	30095	-561
85	30155	60
86	30338	183
87	30451	113
88	30585	134
89	30733	148
90	30892	159
91	31055	163
92	31168	113
93	31906	738
94	31991	85

[Get dates from Release Date History file](#)

From V88, stats are collected from:

[Reactome Statistics Page](#) (automatically generated)

[Reactome Front Page Stats](#) (automated) see <https://download.reactome.org/XX/stats/summary>

[Variant statistics script report](#) (as of V88)

[CT queries on slice](#) (see pre -V88 methods below)

[Stats page](#) [or](#) [Front page](#)

Prior to V88 statistics were generated as follows:

Get data from

Prot = ReferenceGeneProducts whose species = Homo sapiens

netprot = prot - referenceIsoforms whose species = Homo sapiens

complexes = complexes whose species = Homo sapiens

rxn = reactionlikeEvents whose species = Homo sapiens

litRef = literatureReferences

chemicals = referenceMolecules

chemicalDr

newprot = netprot(this release) - netprot(previous release)

EWAS = EWAS whose species = Homo sapiens

DiseaseProt and DiseaseVar: calculated by checking out all geneticallyModifiedResidue instances

pathways = pathways whose species = Homo sapiens

DiseaseRxn=all human reactionlikeEvent disease not null

DiseasePath=all human pathways disease not null

Corona stats - updated for version 76

10 proteins

10 net proteins

152 EWASs

124 reactions (relatedSpecies matches Human SARS coronavirus)

108 complexes (79 purely viral, 29 virus + host)

ces. DiseaseProt is the number of human referenceGeneProducts brought along as referrers and [

DiseaseVar is the total number of geneticallyModifiedResidue instances minus the number that re

fer to non-human referenceGeneProducts

Appendix R5. DOI submission

Identifying pathways needing a new/revised DOI

The CrossRef system is used for DOI submission; read more about Crossref [here](https://www.crossref.org/) at <https://www.crossref.org/>.

A DOI is assigned to a pathway when the pathway (or its contained subevents) have been authored, reviewed, or revised by an external expert. The assigned DOI has the format : 10.3180/R-XXX-XXXXX.Y where R-XXX-XXXXX is the pathway StableIdentifier and Y is the stable identifier version number at the time it is released to the public. When a pathway is updated, the StableIdentifier version number is automatically incremented to reflect the changes*. The DOI for the revised pathway will contain the same StableID but with an incremented stableID version number; thus, different versions of a pathway are associated with different DOIs and CrossRef metadata.

Note that the DOI is assigned to the top-most pathway the contributor authored, reviewed, or revised during a release cycle. During the Reactome release process, the DOI of the relevant pathway instances will be edited to contain "needs DOI" (removing the previous DOI if one existed).

**See documentation on _updateTracker for a description of how changes to Reactome content are tracked.

DOI QA Protocol

A QA protocol monitors that the expected new or updated DOIs are registered in the database and that the list of authors in a DOI record submitted to crossref matches the list of new contributors associated with the corresponding Reactome pathway (or its immediate child event) in the database at that release.

QA protocol (before final slice)

Step 1:

- Query gk_central to retrieve DB_ID for Pathway whose DOI = needs DOI.
- Compare this to Projects with needs DOI "yes" Project management "project" sheet for that release.
- Save final list as: VXX_Pathways_with_new_DOIs

Step 2:

- The DOI suggerter script is run as part of the release QA pipeline and reports new contributors associated with pathways in slice database.
- The managing editor compares the pathways and contributors reported by the script to the contributors listed for those pathways in the Project calendar (authors) and Reviewer management sheet (reviewers).
- Any differences are reconciled with curators. The reconciled list of contributors is kept [here](#).

Step 3: (after final slice)

- The list of DOIs in the final slice (reported by release pipeline) is compared to the reconciled list above.
- The contributors for each pathway are then listed as author in the DOI record to be submitted to crossref.

CrossRef Submission

Every release is submitted as an xml batch file to crossref using the CrossRef xml schema library [here](#). The help system is located [here](https://support.crossref.org/hc/en-us/) <https://support.crossref.org/hc/en-us/>.

For each submission a [template](#) is used, manually adding appropriate versions and release dates, along with each specific module that a DOI has been requested for.

Once the xml file is built out with the appropriate Reactome records, the file is validated using the CrossRef "MetaData Quality Check Tool", located [here](#). Once no schema or xml errors are detected that file is uploaded to <https://doi.crossref.org/servlet/useragent?func=showHome> under the "Submissions" tab. The batch file is identified as type:"Metadata".

The system takes a few minutes to a few hours to incorporate the new DOIs and take them live. An email is issued detailing the success or failure of each submission.

Category

Consistency between pathway diagrams and database content

Proper contributor attribution

Data internal consistency

Missing required experimental evidence

Readability and navigability of pathway diagrams

Verification of external links

Verification of proper manual review of inferred/new/revised data

<u>Code</u>	<u>Name</u>
DT101	DiagramEmpty
DT102	MissingSchemaClass
DT103	ExtraParticipantInDiagram
DT104	DuplicatedReactionParticipants
DT111	UnrecognisedRenderableClass
DT112	UnidentifiedCompartments
DT401	SubpathwaysWithoutParticipants
DT701	ReactionParticipantsMismatch
DT702	DiagramMissingReaction
DT703	ExtraReactionInDiagram
DT106	SchemaClassMismatch
DT109	OverlappingReactions
DT115	NodeAttachmentMismatch
DT116	StoichiometryMismatch
DT117	ParticipantWrongRole
DT105	RenderableClassMismatch
DT107	WronglyPlacedReactionShape
DT113	ReactionShapeMismatch
DT904	PhysicalEntityWrongTranslationalModi
DT108	IsolatedGlyphs
DT110	OverlappingEntities
DT114	ReplacedNodeAttachmentLabel

Description

Detects diagrams without any nodes inside them.

Detects diagrams containing entities without a SchemaClass.

Participants seen in a diagram that shouldn't be there. Mainly t

Detects diagram reactions with duplicate inputs, outputs, catal

A diagram glyph has an unrecognised or unsupported Render

A diagram contains an unidentified compartment (report to the

Detects subpathways that include no diagram participants.

Detects mismatches between the participants of a reaction in a

Detects diagrams with missing reactions compared to how the

Reactions seen in a diagram that shouldn't be there. Mainly th

SchemaClass of a diagram entity does not match the one stor

Detects pairs of reactions which main shapes physically overla

Entities in the diagram that contain different node attachments

Reaction participants where stoichiometry is different to the an

Participants whose role is wrongly displayed in the diagram.

Detects diagrams containing entities with a mismatch in their a

Detects cases where the reaction shapes (Rectangle for transi

Detects reaction categories that differ from the expected ones

PhysicalEntity instances with a potentially wrongly annotated tr

Iterates over all Edges and Links and identifies the isolated gly

Detects pairs of entities that physically overlap in the diagram.

Fixes the node attachments labels based on the database ann

Category

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Readability and navigability of ELV pathway diagrams

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Consistency between pathway diagrams and database content

Readability and navigability of ELV pathway diagrams/Consistency between pathway diagrams and databa

Readability and navigability of ELV pathway diagrams

Consistency between pathway diagrams and database content

Code	Name
GT003	DatabasIdentifierWithoutIdentifier
GT005	PathwaysWithoutEvents
GT006	EwasWithoutReferenceEntity
GT007	EntitiesWithoutStd
GT009	ComplexWithoutComponents
GT012	SimpleEntityWithoutReferenceEntity
GT013	ReferenceEntityWithoutIdentifier
GT022	PhysicalEntityWithoutCompartment
GT101	CompartmentWithCyclicSurroundedByAndInstanceOf
GT102	FiguresURLInconsistence
GT004	EntriesWithoutDisplayName
GT010	EntitySetWithoutMemberOrCandidate
GT011	PolymerWithoutRepeatedUnit
GT015	CatalystActivityWithoutPhysicalEntity
GT016	CatalystActivityWithoutActivity
GT017	NOT_FailedReactionsWithoutOutputs
GT019	PublicationsWithoutAuthor
GT021	RegulationsWithoutRegulatedEntityOrRegulator
GT024	ReferenceDatabaseWithoutUrls
GT028	HasMemberAndHasCandidatePointToSameEntry
GT029	ReactionsLikeEventWithoutInput
GT036	EventsWithoutCompartment
GT037	LiteratureReferenceRelationshipDuplication
GT038	HasModifiedResidueRelationshipDuplication
GT040	HasMemberRelationshipDuplication
GT043	HasCandidateRelationshipDuplication
GT044	HasEventRelationshipDuplication
GT050	DuplicatedLiteratureReferences
GT055	DuplicatedCuratedComplexes
GT057	ComplexesWithOnlyOneComponent
GT058	ComplexesWhereCompartmentDoesNotMatchWithAnyOfThe
GT059	DuplicatedCandidateSets
GT062	DuplicatedReferenceEntities
GT064	DuplicatedEntitySets
GT070	ReactionsWithoutRegulatorWithMoreCompartmentsThanItsP
GT071	ReactionsWithOnlyOneInputAndOutputWhereSchemaClassC
GT090	CatalystActivityCompartmentDoesNotMatchReactionCompar
GT091	ReactionsWithoutLiteratureReference
GT092	PotentialTranslocationReactionChangesParticipantsSchemaC
GT100	DiseasePathwayWithoutNormalPathwayOrNormalPathwayHa
GT001	DatabaseObjectsWithSelfLoops
GT020	OpenSetsWithoutReferenceEntity
GT023	DatabasIdentifierWithoutReferenceDatabase

GT025	EntriesWithCyclicInferredToRelations
GT026	EventsWithCyclicPrecedingEvents
GT027	EntriesWithOtherCyclicRelations
GT030	PhysicalEntitiesWithMoreThanOneCompartment
GT031	CatalystActivityWherePhysicalEntityAndActiveUnitPointToCor
GT032	PrecedingEventOrReverseReactionOrHasEventPointToSame
GT035	CrossReferenceRelationshipDuplication
GT039	PrecedingEventRelationshipDuplication
GT041	SummationRelationshipDuplication
GT042	PsiModRelationshipDuplication
GT047	OrphanEvents
GT051	PersonsWithSameORCIDAndProject
GT053	PrecedingEventOutputsNotUsedInReaction
GT061	EntitySetsWithOnlyOneMember
GT065	EntitySetsWithRepeatedMembers
GT002	PersonWithoutProperName
GT014	InstanceEditWithoutAuthor
GT018	DatabaseObjectsWithoutCreated
GT033	OtherRelationsThatPointToTheSameEntry
GT034	ModifiedRelationshipDuplication
GT045	OtherRelationshipDuplication
GT048	InstanceEditCreatesInstanceEdit
GT049	InstanceEditModifiesInstanceEdit
GT052	PersonsWithSameNameAndProject

Description	Category	Comment
DatabasIdentifier class instances where the identifier slot is empty	Data internal consistency	
Pathway class instances where the hasEvent slot is empty	Data internal consistency	
EWAS class instances where the referenceEntity slot is empty	Data internal consistency	
Event and PhysicalEntity class instances where stableIdentifier slot is empty	Data internal consistency	
Complexes instances where the hasComponent slot is empty	Data internal consistency	
SimpleEntity class instances where referenceEntity slot is empty	Data internal consistency	
ReferenceEntity class instances where the identifier slot is empty	Data internal consistency	
PhysicalEntity class instances where the compartment slot is empty	Data internal consistency	
Compartments with cyclic relationships in the surroundedBy, composedOf, or contains slot	Data internal consistency	
Figures (ehld/static) with potential mismatch or missing extension	Data internal consistency	
The instance's displayName slot is empty	Data internal consistency	
EntitySet instances where both hasMember and hasCandidate slots are empty	Data internal consistency	
Polymer class instances where the repeatedUnit slot is empty	Data internal consistency	
CatalystActivity class instances where the physicalEntity slot is empty	Data internal consistency	
CatalystActivity class instances where the activity slot is empty	Data internal consistency	
ReactionLikeEvent class instance (excluding FailedReaction class instances)	Data internal consistency	
Publication class instances where the author slot is empty	Data internal consistency	
Regulation class instances where the regulatedEntity or the regulator slot is empty	Data internal consistency	
ReferenceDatabase class instances where accessUrl slot or url slot is empty	Data internal consistency	
CandidateSet class instances where at least one instance in the hasCandidate slot is empty	Data internal consistency	
ReactionLikeEvent instances where the input slot is empty	Data internal consistency	
Event class instances where the compartment slot is empty	Data internal consistency	
DatabaseObject class instances where the literatureReference slot is empty	Data internal consistency	
DatabaseObject class instances where the hasModifiedResidue slot is empty	Data internal consistency	
DatabaseObject class instances where the hasMember slot contains empty list	Data internal consistency	
CandidateSet class instances where the hasCandidate slot contains empty list	Data internal consistency	
Event class instances where the hasEvent slot contains duplicated entries	Data internal consistency	
Different instances of the LiteratureReference class with the same PMID	Data internal consistency	
Two different instances of the class Complex that have the same input and output	Data internal consistency	
Complex class instances with only one entry in the hasComponent slot	Data internal consistency	
Complexes where the compartment does not match with any of the participants	Data internal consistency	
Two different instances of the class CandidateSet that have the same identifier	Data internal consistency	
Two different instances of the class ReferenceEntity that have the same identifier	Data internal consistency	
Two different instances of the class EntitySet that have the same identifier	Data internal consistency	
Reactions without regulator with more compartments than its participants	Data internal consistency	
Reactions with only one input and output where schemaClass does not match	Data internal consistency	
CatalystActivity class instances where the compartment slot does not match	Data internal consistency	
Reactions with no LiteratureReference class instances in any of the participants	Missing required experimental evidence, Data internal consistency	
Potential translocation Reaction changes participants schemaClass	Data internal consistency	
Disease Pathway instances without normalPathway or NormalPathway	Data internal consistency	
The instance contains itself in any of the slots	Data internal consistency	
OpenSet class instances where the referenceEntity slot is empty	Data internal consistency	
DatabasIdentifier class instances where the referenceDatabase slot is empty	Data internal consistency	

When DatabaseObject class instance (A) is used to infer another instance (B) in the hierarchy Data internal consistency

When an Event class instance (A) contains another instance (B) in the hierarchy Data internal consistency

Same concept as the two above but checking for other slots (and not just the compartment slot) Data internal consistency

PhysicalEntity class instances where the compartment slot contains another PhysicalEntity Data internal consistency

CatalystActivity class instances where the activity and activeUnit slot contain the same Activity Data internal consistency

PrecedingEvent or ReverseReaction or HasEvent point to same Event Data internal consistency

DatabaseObject class instances where the crossReference slot contains another DatabaseObject Data internal consistency

DatabaseObject class instances where the precedingEvent slot contains another Event Data internal consistency

DatabaseObject class instances where the summation slot contains another DatabaseObject Data internal consistency

DatabaseObject class instances where the psiMod slot contains duplicate values Data internal consistency

Events that cannot be reached through the events hierarchy Data internal consistency

Different instances of the Person class with the same ORCID identifier Contributor attribution check

ReactionLikeEvent (A) annotated as preceding of another one (B) with the same precedingEvent Data internal consistency

EntitySet class instances (excluding CandidateSet and OpenSet) that have the same precedingEvent Data internal consistency

EntitySet class instances (excluding CandidateSet) where the same precedingEvent Data internal consistency

Surname is empty or the Firstname and the Initial are also empty Data internal consistency

InstanceEdit class instances where the author slot is empty Data internal consistency

DatabaseObject class instances (excluding InstanceEdit, DatabaseObject, and ReactionLikeEvent) Data internal consistency

Other relations that point to the same entry Data internal consistency

DatabaseObject class instances where the modified slot contains duplicate values Data internal consistency

DatabaseObject class instances with duplicated instances in a multi-valued slot Data internal consistency

InstanceEdit class instances that are created by other InstanceEdit class instances Data internal consistency

InstanceEdit class instances that are modified by other InstanceEdit class instances Data internal consistency

Different instances of the Person class with the same name and project Data internal consistency

Data internal consistency

Verifying manual review of inferred/new/revised

Readability and navigability of ELV pathway diagrams

Consistency between pathway diagrams and data

Missing required experimental evidence, Data integrity

Contributor attribution check

Verification of external links

Code	Name	Descriptions
CL1	Complex Components Species Mismatch	Complex Components Species
CL2	Deleted Objects In Diagram	Deleted Objects In Diagram
CL3	Diagram EWAS Modification Mismatch	Diagram EWAS Modification M
CL4	Diagram Subpathway Undiagrammed Re	Diagram Subpathway Undiagr
CL5	Diagram Unrepresented Reactions	Diagram Unrepresented React
CL6	Diagram Unsynchronized Reactions	Diagram Unsynchronized Rea
CL7	Diagram With Missing Reactions	Diagram With Missing Reactio
CL8	Diagram With Unconnected Entities	Diagram With Unconnected Er
CL9	Disease Entity Inconsistent	Verify
CL10	EntitySet Members Species Mismatch	EntitySet Members Species Mi
CL11	Extra Compartments In Entity Set Or Men	Extra Compartments In Entity :
CL12	Instances Without StableIdentifier	Instances Without StableIdenti
CL13	Mandatory Attributes	Mandatory Attributes
CL14	Normal Entity Inconsistent	Verify
CL15	Pathway Without Diagram	Pathway Without Diagram
CL16	Reaction Input Output Imbalance	Reaction Input Output Imbalan

Category	Comment
Data internal consistency	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Consistency between pathway diagrams and database content	
Data internal consistency	

Code	Name
R1	Attribute Has Multiple Values
R2	Attribute Has Only One Value
R3	Attribute Value Duplication
R4	Attribute Value Missing
R5	CatalystActivity PhysicalEntity ActivityUnit Refers T
R6	Chimerism Reference Constraint Violations
R7	Complexes That Should Have Homo sapiens Spec
R8	CoV-2 Entities With CoV-1 Species Or DisplayNarr
R9	CoV-2 Infection Pathway Events With Summation /
R10	DatabaseObject With Self Loop
R11	Diagram Compartment Label Missing
R12	Diagram Disease Color
R13	Diagram Drug Color
R14	Diagram Duplicate Reaction Participants
R15	Diagram Empty
R16	Diagram Extra ReactionLikeEvents
R17	Diagram Overlapping Entities
R18	Diagram Overlapping Reactions
R19	Diagram Reactions With Participant Overlapping H
R20	Diagram Wrong Renderable Class
R21	EHL D Subpathway Change
R22	Entities With CoV Species Without Corresponding
R23	Events With Electronic Evidence Types
R24	FailedReaction Has Output
R25	Human Reactions Without Disease And Have Nonl
R26	Inferred Modified Residues
R27	InferredFrom Used In Other Attribute
R28	Instance Duplication
R29	Missing Editorial Attribute
R30	Multiple Attributes Cross Classes Missing Simultan
R31	Multiple Attributes Missing Simultaneously
R32	New Event Inconsistent
R33	NonHuman Events Not Manually Inferred
R34	NonHuman Reactions Containing Human Physicall
R35	One Hop Circular Reference
R36	Orcid Crossreference
R37	Orphan Events
R38	PhysicalEntity With More Than One Compartment
R39	PhysicalEntity Without Species Components With :
R40	Reaction Single Input Output Schema Not Matched
R41	ReactionlikeEvent Not Failed With Normal Without
R42	ReactionlikeEvent Regulations Not Reviewed
R43	Reference Database Access URL

R44	Review Status
R45	SimpleEntity Has Species
R46	Species In Preceding Event
R47	Stable Identifier
R48	Two Attributes Refer To Same Instance
R49	Two Attributes Refer To Same Instance
R50	Values Not Unique

Description	Category
Attribute Has Multiple Values where should only have	Data internal consistency
Attribute Has Only One Value where should have >1	Data internal consistency
Attribute Value Duplication	Data internal consistency
CatalystActivity PhysicalEntity ActivityUnit Refers To	Data internal consistency
ActivityUnit only used if more specific than the P.E	Data internal consistency
Verifies Instances with IsChimeric attribute have other	Data internal consistency
Complexes That Should Have Homo sapiens Species	Data internal consistency
CoV-2 Entities With CoV-1 Species Or DisplayName	Data internal consistency
CoV-2 Infection Pathway Events With unmodified Sur	Verifying manual review of inferred/new/revised data
Class instance should not have instances that refering	Readability and navigability of ELV pathway diagrams
Diagram Compartment Label Missing	Readability and navigability of ELV pathway diagrams
Disease entities in diagram must be updated	Consistency between pathway diagrams and database conte
Drug entities in diagram must be updated	Consistency between pathway diagrams and database conte
Duplicate Reaction Participants in a diagram	Consistency between pathway diagrams and database conte
Diagram Empty	Consistency between pathway diagrams and database conte
Extra ReactionLikeEvents in diagram that are not in th	Consistency between pathway diagrams and database conte
Diagram with Overlapping Reactions	Consistency between pathway diagrams and database conte
Diagram Reactions With Participant Overlapping Rea	Readability and navigability of ELV pathway diagrams
Diagram has reactions with participant overlapping the	Readability and navigability of ELV pathway diagrams
Diagram entity has the wrong renderable class	Consistency between pathway diagrams and database conte
Pathways associated with an EHLD diagram that have	Consistency between pathway diagrams and database conte
Entities With CoV Species without corresponding dise	Data internal consistency
Events that have been inferred from electronic inferer	Data internal consistency
FailedReaction with an Output	Data internal consistency
Human reactions without a disease attribute but havii	Data internal consistency
Modified residues in slice database that still contain [I]	Data internal consistency
InferredFrom Used In Other Attribute	Verifying manual review of inferred/new/revised data
Instance Duplication	Data internal consistency
Missing Editorial Attribute	Verifying manual review of inferred/new/revised data
Missing attributes in indicated classes	Missing required experimental evidence, Data internal consi
Certain attributes for a given class should not be miss	Missing required experimental evidence, Data internal consi
?	Data internal consistency
Non-human events under Human hierarchy should ha	Data internal consistency
Non-human events NOT under Human hierarchy shou	Data internal consistency
Attribute assignment in instance creates an inaccurat	Data internal consistency
?	Contributor attribution check
Event is set as doRelease =True but not connected to	Data internal consistency
PhysicalEntity With More Than One Compartment	Data internal consistency
PhysicalEntity species is null but components species	Data internal consistency
Molecule type is different in the input vs. output	Data internal consistency
ReactionlikeEvent Not Failed but has NormalEvent ar	Data internal consistency
No longer needed	Verifying manual review of inferred/new/revised data
?	Verification of external links

ReviewStatus setting inconsisted with reviewDate / la: Verification of ReviewStatus accuracy

SimpleEntity Has Species Data internal consistency

Inconsistent Species In Preceding Event Data internal consistency

No longer needed (?) Data internal consistency

Two attributes of one instance are identical but shoulc Data internal consistency

Two attributes of one instance are identical but shoulc Data internal consistency

Two attributes of one instance are identical but shoulc Data internal consistency

Comment

ent
ent
ent
ent
ent
ent

Deprecate ...The script (I believe renderable_entity_qa.pl) is producing incorrect flagged events Guanming reco
ent

stency
stency

deprecate

Add to weeklyQA

deprecate

reports with duplicate name difference instance count

reports with duplicate name difference instance count

Code	Name
R1	Attribute Has Multiple Values
R2	Attribute Has Only One Value
R3	Attribute Value Duplication
R4	Attribute Value Missing
R5	CatalystActivity PhysicalEntity ActivityUnit Refers To Same Complex
R6	Chimerism Reference Constraint Violations
R7	Complexes That Should Have Homo sapiens Species
R8	CoV-2 Entities With CoV-1 Species Or DisplayName
R9	CoV-2 Infection Pathway Events With Summation And Literature Refer
R10	DatabaseObject With Self Loop
R11	Diagram Compartment Label Missing
R12	Diagram Disease Color
R13	Diagram Drug Color
R14	Diagram Duplicate Reaction Participants
R15	Diagram Empty
R16	Diagram Extra ReactionLikeEvents
R17	Diagram Overlapping Entities
R18	Diagram Overlapping Reactions
R19	Diagram Reactions With Participant Overlapping Hub
R20	Diagram Wrong Renderable Class
R21	EHLN Subpathway Change
R22	Entities With CoV Species Without Corresponding Disease
R23	Events With Electronic Evidence Types
R24	FailedReaction Has Output
R25	Human Reactions Without Disease And Have NonHuman PhysicalEnti
R26	Inferred Modified Residues
R27	InferredFrom Used In Other Attribute
R28	Instance Duplication
R29	Missing Editorial Attribute
R30	Multiple Attributes Cross Classes Missing Simultaneously
R31	Multiple Attributes Missing Simultaneously
R32	New Event Inconsistent
R33	NonHuman Events Not Manually Inferred
R34	NonHuman Reactions Containing Human PhysicalEntities
R35	One Hop Circular Reference
R36	Orcid Crossreference
R37	Orphan Events
R38	PhysicalEntity With More Than One Compartment
R39	PhysicalEntity Without Species Components With Species
R40	Reaction Single Input Output Schema Not Matched
R41	ReactionlikeEvent Not Failed With Normal Without Disease
R42	ReactionlikeEvent Regulations Not Reviewed
R43	Reference Database Access URL

R44	SimpleEntity Has Species
R45	Species In Preceding Event
R46	Stable Identifier
R47	Two Attributes Refer To Same Instance
R48 or R49 ?	
R50	Values Not Unique

Description (where clarification needed)

Attribute Has Multiple Values where should on
Attribute Has Only One Value where should ha
Attribute Value Duplication

CatalystActivity PhysicalEntity ActivityUnit Refe

ActivityUnit only used if more specific than the

Verifies Instances with IsChimeric attribute ha

Complexes That Should Have Homo sapiens :

CoV-2 Entities With CoV-1 Species Or Display

CoV-2 Infection Pathway Events With unmodif

Class instance should not have instances that

Diagram Compartment Label Missing

Disease entities in diagram must be updated

Drug entities in diagram must be updated

Duplicate Reaction Participants in a diagram

Diagram Empty

Extra ReactionLikeEvents in diagram that are

Diagram with Overlapping Reactions

Diagram Reactions With Participant Overlappi

Diagram has reactions with participant overlap

Diagram entity has the wrong renderable class

Pathways associated with an EHLD diagram th

Entities With CoV Species without correspond

Events that have been inferred from electronic

FailedReaction with an Output

Human reactions without a disease attribute b

Modified residues in slice database that still cc

InferredFrom Used In Other Attribute

Instance Duplication

Missing Editorial Attribute

Missing attributes in indicated classes

Certain attributes for a given class should not l

?

Non-human events under Human hierarchy sh

Non-human events NOT under Human hierarc

Attribute assignment in instance creates an in

?

Event is set as doRelease =True but not conn

PhysicalEntity With More Than One Compartn

PhysicalEntity species is null but components :

Molecule type is different in the input vs. outpu

ReactionlikeEvent Not Failed but has NormalE

No longer needed

?

SimpleEntity Has Species

Inconsistent Species In Preceding Event

No longer needed (?)

Two attributes of one instance are identical bu

?

Two attributes of one instance are identical bu

Category	Comment
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Verifying manual review of inferred/new/revised data	
Readability and navigability of ELV pathway diagrams	
Readability and navigability of ELV pathway diagrams	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Consistency between pathway diagrams and database content	
Readability and navigability of ELV pathway diagrams	
Readability and navigability of ELV pathway diagrams	
Consistency between pathway diagrams and database content	Deprecate ...The script (I believe <u>rend</u>
Consistency between pathway diagrams and database content	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Verifying manual review of inferred/new/revised data	
Data internal consistency	
Verifying manual review of inferred/new/revised data	
Missing required experimental evidence, Data internal consistency	
Missing required experimental evidence, Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Contributor attribution check	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Data internal consistency	
Verifying manual review of inferred/new/revised data	deprecate
Verification of external links	

Data internal consistency

Data internal consistency

Data internal consistency

Data internal consistency

Data internal consistency

Data internal consistency

deprecate

Code	Name	Description	Category
DT101	DiagramEmpty	Detects diagrams without any nodes inside them.	Consistency between pathway diagrams and database content
DT102	MissingSchemaClass	Detects diagrams containing entities without a SchemaClass.	Consistency between pathway diagrams and database content
DT103	ExtraParticipantInDiagram	Participants seen in a diagram that shouldn't be there. Mainly the reason is because they are annotated in the database.	Consistency between pathway diagrams and database content
DT104	DuplicatedReactionParticipants	Detects diagram reactions with duplicate inputs, outputs, catalysts, activators, inhibitors.	Consistency between pathway diagrams and database content
DT111	UnrecognisedRenderableClass	A diagram glyph has an unrecognised or unsupported RenderableClass (Please report to developers)	Consistency between pathway diagrams and database content
DT112	UnidentifiedCompartments	A diagram contains an unidentified compartment (report to the developers list)	Consistency between pathway diagrams and database content
DT401	SubpathwaysWithoutParticipants	Detects subpathways that include no diagram participants.	Consistency between pathway diagrams and database content
DT701	ReactionParticipantsMismatch	Detects mismatches between the participants of a reaction in a diagram and the database.	Consistency between pathway diagrams and database content
DT702	DiagramMissingReaction	Detects diagrams with missing reactions compared to how they are annotated in the database.	Consistency between pathway diagrams and database content
DT703	ExtraReactionInDiagram	Reactions seen in a diagram that shouldn't be there. Mainly the reason is because they are annotated in the database.	Consistency between pathway diagrams and database content
DT106	SchemaClassMismatch	SchemaClass of a diagram entity does not match the one stored in the database.	Consistency between pathway diagrams and database content
DT109	OverlappingReactions	Detects pairs of reactions which main shapes physically overlap in the diagram.	Readability and navigability of ELV pathway diagrams
DT115	NodeAttachmentMismatch	Entities in the diagram that contain different node attachments to those annotated in the database.	Consistency between pathway diagrams and database content
DT116	StoichiometryMismatch	Reaction participants where stoichiometry is different to the annotated in the database.	Consistency between pathway diagrams and database content
DT117	ParticipantWrongRole	Participants whose role is wrongly displayed in the diagram.	Consistency between pathway diagrams and database content
DT105	RenderableClassMismatch	Detects diagrams containing entities with a mismatch in their annotated RenderableClass.	Consistency between pathway diagrams and database content
DT107	WronglyPlacedReactionShape	Detects cases where the reaction shapes (Rectangle for transition, Circle for binding) are placed incorrectly.	Consistency between pathway diagrams and database content
DT113	ReactionShapeMismatch	Detects reaction categories that differ from the expected ones based on the annotation.	Consistency between pathway diagrams and database content
DT904	PhysicalEntityWrongTranslationalModification	PhysicalEntity instances with a potentially wrongly annotated translational modification.	Consistency between pathway diagrams and database content
DT108	IsolatedGlyphs	Iterates over all Edges and Links and identifies the isolated glyphs. The method could be used to detect orphaned glyphs.	Readability and navigability of ELV pathway diagrams/Consistency between pathway diagrams and database content
DT110	OverlappingEntities	Detects pairs of entities that physically overlap in the diagram.	Readability and navigability of ELV pathway diagrams
DT114	ReplacedNodeAttachmentLabel	Fixes the node attachment labels based on the database annotation and reports inconsistencies.	Consistency between pathway diagrams and database content
GT003	DatabaseIdentifierWithoutIdentifier	DatabaseIdentifier class instances where the identifier slot is empty	Data internal consistency
GT005	PathwaysWithoutEvents	Pathway class instances where the hasEvent slot is empty	Data internal consistency
GT006	EwasWithoutReferenceEntity	EWAS class instances where the referenceEntity slot is empty	Data internal consistency
GT007	EntitiesWithoutStd	Event and PhysicalEntity class instances where stabeldentifier slot is empty	Data internal consistency
GT009	ComplexWithoutComponents	Complexes instances where the hasComponent slot is empty	Data internal consistency
GT012	SimpleEntityWithoutReferenceEntity	SimpleEntity class instances where referenceEntity slot is empty	Data internal consistency
GT013	ReferenceEntityWithoutIdentifier	ReferenceEntity class instances where the identifier slot is empty	Data internal consistency
GT022	PhysicalEntityWithoutCompartment	PhysicalEntity class instances where the compartment slot is empty	Data internal consistency
GT101	CompartmentWithCyclicSurroundedByAndIn	Compartments with cyclic relationships in the surroundedBy, componentOf and/or in slots.	Data internal consistency
GT102	FiguresURLInconsistence	Figures (ehid/static) with potential mismatch or missing extension	Data internal consistency
GT004	EntriesWithoutDisplayName	The instance's displayName slot is empty	Data internal consistency
GT010	EntitySetWithoutMemberOrCandidate	Entity/Set instances where both hasMember and hasCandidate slots are empty	Data internal consistency
GT011	PolymerWithoutRepeatedUnit	Polymer class instances where the repeatedUnit slot is empty	Data internal consistency
GT015	CatalystActivityWithoutPhysicalEntity	CatalystActivity class instances where the physicalEntity slot is empty	Data internal consistency
GT016	CatalystActivityWithoutActivity	CatalystActivity class instances where the activity slot is empty	Data internal consistency
GT017	NOT_FailedReactionsWithoutOutputs	ReactionLikeEvent class instance (excluding FailedReaction class instances) where the output slot is empty	Data internal consistency
GT019	PublicationsWithoutAuthor	Publication class instances where the author slot is empty	Data internal consistency
GT021	RegulationsWithoutRegulatedEntityOrRegulator	Regulation class instances where the regulatedEntity or the regulator slots is empty	Data internal consistency
GT024	ReferenceDatabaseWithoutUris	ReferenceDatabase class instances where accessUri slot or uri slot are empty (either)	Data internal consistency
GT028	HasMemberAndHasCandidatePointToSame	CandidateSet class instances where at least one instance in the hasCandidate slot points to the same instance as the hasMember slot	Data internal consistency
GT029	ReactionsLikeEventWithoutInput	ReactionLikeEvent instances where the input slot is empty	Data internal consistency
GT036	EventsWithoutCompartment	Event class instances where the compartment slot is empty	Data internal consistency
GT037	LiteratureReferenceRelationshipDuplication	DatabaseObject class instances where the literatureReference slot contains duplicated entries	Data internal consistency
GT038	HasModifiedResidueRelationshipDuplication	DatabaseObject class instances where the hasModifiedResidue slot contains duplicated entries	Data internal consistency
GT040	HasMemberRelationshipDuplication	DatabaseObject class instances where the hasMember slot contains duplicated entries	Data internal consistency
GT043	HasCandidateRelationshipDuplication	CandidateSet class instances where the hasCandidate slot contains duplicated entries	Data internal consistency
GT044	HasEventRelationshipDuplication	Event class instances where the hasEvent slot contains duplicated entries	Data internal consistency
GT050	DuplicatedLiteratureReferences	Different instances of the LiteratureReference class with the same PubMed identifiers	Data internal consistency
GT055	DuplicatedCuratedComplexes	Two different instances of the class Complex that have the same instances in the hasComponent slot	Data internal consistency
GT057	ComplexesWithOnlyOneComponent	Complex class instances with only one entry in the hasComponent slot	Data internal consistency
GT058	ComplexesWhereCompartmentDoesNotMatch	Complexes where the compartment does not match with any of the participants	Data internal consistency
GT059	DuplicatedCandidateSets	Two different instances of the class CandidateSet that have the same instances in the hasMember slot	Data internal consistency
GT062	DuplicatedReferenceEntities	Two different instances of the class ReferenceEntity that have the same content	Data internal consistency
GT064	DuplicatedEntitySets	Two different instances of the class EntitySet that have the same content	Data internal consistency
GT070	ReactionsWithoutRegulatorWithMoreCompartments	Reactions without regulator with more compartments than its participants	Data internal consistency
GT071	ReactionsWithOnlyOneInputAndOutputWhere	Reactions with only one input and output where schemaClass does not match with the database	Data internal consistency
GT090	CatalystActivityCompartmentDoesNotMatch	CatalystActivity class instances where the compartment slot does not match with the database	Data internal consistency
GT091	ReactionsWithoutLiteratureReference	Reactions with no LiteratureReference class instances in any of the possible locations	Missing required experimental evidence, Data internal consistency
GT092	PotentialTranslocationReactionChangesParticipant	Potential translocation Reaction changes participants schemaClass	Data internal consistency
GT100	DiseasePathwayWithoutNormalPathwayOrNormalPathwayWithoutDisease	Disease Pathway instances without normalPathway or NormalPathway has no disease	Data internal consistency
GT001	DatabaseObjectsWithSelfLoops	The instance contains itself in any of the slots	Data internal consistency
GT020	OpenSetsWithoutReferenceEntity	OpenSet class instances where the referenceEntity slot is empty	Data internal consistency
GT023	DatabaseIdentifierWithoutReferenceDatabase	DatabaseIdentifier class instances where the referenceDatabase slot is empty	Data internal consistency
GT025	EntriesWithCyclicInferredToRelations	When DatabaseObject class instance (A) is used to infer another instance (B), if A is inferred to B, then B is also inferred to A	Data internal consistency
GT026	EventsWithCyclicPrecedingEvents	When an Event class instance (A) contains another instance (B) in the preceding event slot, then B must precede A	Data internal consistency
GT027	EntriesWithOtherCyclicRelations	Same concept as the two above but checking for other slots (and not focusing on the preceding event slot)	Data internal consistency
GT030	PhysicalEntitiesWithMoreThanOneCompartment	PhysicalEntity class instances where the compartment slot contains more than one compartment	Data internal consistency
GT031	CatalystActivityWherePhysicalEntityAndActiveSiteOverlap	CatalystActivity class instances where the activity and activeUnit slots point to the same instance	Data internal consistency
GT032	PrecedingEventOrReverseReactionOrHasEvent	PrecedingEvent or ReverseReaction or HasEvent point to same Event instance	Data internal consistency
GT035	CrossReferenceRelationshipDuplication	DatabaseObject class instances where the crossReference slot contains duplicated entries	Data internal consistency
GT039	PrecedingEventRelationshipDuplication	DatabaseObject class instances where the precedingEvent slot contains duplicated entries	Data internal consistency
GT041	SummationRelationshipDuplication	DatabaseObject class instances where the summation slot contains duplicated entries	Data internal consistency
GT042	PsiModRelationshipDuplication	DatabaseObject class instances where the psiMod slot contains duplicated entries	Data internal consistency
GT047	OrphanEvents	Events that cannot be reached through the events hierarchy	Data internal consistency

GT051	PersonsWithSameORCIDAndProject	Different instances of the Person class with the same ORCID identifier (and project)	Contributor attribution check
GT063	PrecedingEventOrUnitInReaction	ReactionLikeEvent (A) annotated as preceding of another one (B) where none are	Data internal consistency
GT081	EntitySetsWithOnlyOneMember	EntitySet class instances (excluding CandidateSet and OpenSet) that has only one	Data internal consistency
GT065	EntitySetsWithRepeatedMembers	EntitySet class instances (excluding CandidateSet) where the same PhysicalEntity	Data internal consistency
GT002	PersonWithoutProperName	Surname is empty or the Firstname and the Initial are also empty	Data internal consistency
GT014	InstanceEditWithoutAuthor	InstanceEdit class instances where the author slot is empty	Data internal consistency
GT018	DatabaseObjectsWithoutCreated	DatabaseObject class instances (excluding InstanceEdit, DatabaseIdentifier, Taxo	Data internal consistency
GT033	OtherRelationsThatPointToTheSameEntry	Other relations that point to the same entry	Data internal consistency
GT034	ModifiedRelationshipDuplication	DatabaseObject class instances where the modified slot contains duplicated entrie	Data internal consistency
GT045	OtherRelationshipDuplication	DatabaseObject class instances with duplicated instances in a multivalued slot (or	Data internal consistency
GT048	InstanceEditCreatesInstanceEdit	InstanceEdit class instances that are created by other InstanceEdit class instanc	Data internal consistency
GT049	InstanceEditModifiesInstanceEdit	InstanceEdit class instances that are modified by other InstanceEdit class instanc	Data internal consistency
GT052	PersonsWithSameNameAndProject	Different instances of the Person class with the same name and project	Data internal consistency
R1	Attribute Has Multiple Values	Attribute Has Multiple Values where should only have one	Data internal consistency
R2	Attribute Has Only One Value	Attribute Has Only One Value where should have >1	Data internal consistency
R3	Attribute Value Duplication	Attribute Value Duplication	Data internal consistency
R4	Attribute Value Missing	CatalystActivityPhysicalEntity ActivityUnit Refers To Same Complex	Data internal consistency
R5	CatalystActivity PhysicalEntity ActivityUnit Ref	ActivityUnit only used if more specific than the P.E	Data internal consistency
R6	Chimerism Reference Constraint Violations	Verifies instances with IsChimeric attribute have other expected attributes	Data internal consistency
R7	Complexes That Should Have Homo sapiens	Complexes That Should Have Homo sapiens Species	Data internal consistency
R8	CoV-2 Entities With CoV-1 Species Or Display Name	CoV-2 Entities With CoV-1 Species Or Display Name	Data internal consistency
R9	CoV-2 Infection Pathway Events With Summ	CoV-2 Infection Pathway Events With unmodified Summation And Literature Refe	Verifying manual review of inferred/new/revised data
R10	DatabaseObject With Self Loop	Class instance should not have instances that referring to self	Readability and navigability of ELV pathway diagrams
R11	Diagram Compartment Label Missing	Diagram Compartment Label Missing	Readability and navigability of ELV pathway diagrams
R12	Diagram Disease Color	Disease entities in diagram must be updated	Consistency between pathway diagrams and database content
R13	Diagram Drug Color	Drug entities in diagram must be updated	Consistency between pathway diagrams and database content
R14	Diagram Duplicate Reaction Participants	Duplicate Reaction Participants in a diagram	Consistency between pathway diagrams and database content
R15	Diagram Empty	Diagram Empty	Consistency between pathway diagrams and database content
R16	Diagram Extra ReactionLikeEvents	Extra ReactionLikeEvents in diagram that are not in the hierarchy in database	Consistency between pathway diagrams and database content
R17	Diagram Overlapping Entitles	Diagram with Overlapping Reactions	Consistency between pathway diagrams and database content
R18	Diagram Overlap Reactions	Diagram with Reactions with Participant Overlapping Reaction Hub	Readability and navigability of ELV pathway diagrams
R19	Diagram Reactions With Participant Overlap	Diagram has reactions with participant overlapping the hub	Readability and navigability of ELV pathway diagrams
R20	Diagram Wrong Renderable Class	Diagram entity has the wrong renderable class	Consistency between pathway diagrams and database content
R21	EHLID Subpathway Change	Pathways associated with an EHLID diagram that have had a hierarchy structure c	Consistency between pathway diagrams and database content
R22	Entities With CoV Species Without Correspond	Entities With CoV Species without corresponding disease tag	Data internal consistency
R23	Events With Electronic Evidence Types	Events that have been inferred from electronic inference during orthoprojection c	Data internal consistency
R24	FailedReaction Has Output	FailedReaction with an Output	Data internal consistency
R25	Human Reactions Without Disease And Have	Human reactions without a disease attribute but having Non Human PhysicalEntit	Data internal consistency
R26	Inferred Modified Residues	Modified residues in slice database that still contain [INFERRED] tag and need to	Data internal consistency
R27	InferredFrom Used In Other Attribute	InferredFrom Used In Other Attribute	Verifying manual review of inferred/new/revised data
R28	Instance Duplication	Instance Duplication	Data internal consistency
R29	Missing Editorial Attribute	Missing Editorial Attribute	Verifying manual review of inferred/new/revised data
R30	Multiple Attributes Cross Classes Missing Sim	Missing attributes in indicated classes	Missing required experimental evidence, Data internal consistency
R31	Multiple Attributes Missing Simultaneously	Certain attributes for a given class should not be missing simultaneously, e.g if lit	Missing required experimental evidence, Data internal consistency
R32	New Event Inconsistent	New event missing expected attributes	Data internal consistency
R33	NonHuman Events Not Manually Inferred	Non-human events under Human hierarchy should have species=H.s., unless use	Data internal consistency
R34	NonHuman Reactions Containing Human Pat	non-human events NOT under Human hierarchy should not have entities with spe	Data internal consistency
R35	One Hop Circular Reference	Attribute assignment in instance creates an inaccurate circular reference	Data internal consistency
R36	Oxid Crossreference	ORCID inconsistency	Contributor attribution check
R37	Orphan Events	Event is set as doRelease =True but not connected to the main event hierarchy	Data internal consistency
R38	PhysicalEntity With More Than One Compart	PhysicalEntity With More Than One Compartment.	Data internal consistency
R39	PhysicalEntity Without Species Components	PhysicalEntity species is null but components species are not null	Data internal consistency
R40	Reaction Single Input Output Schema Not Ma	Molecule type is different in the input vs. output	Data internal consistency
R41	ReactionlikeEvent Not Failed With Normal W	ReactionlikeEvent Not Failed but has NormalEvent and not Disease	Data internal consistency
R42	ReactionlikeEvent Regulations Not Reviewed	No longer needed	Verifying manual review of inferred/new/revised data
R43	Reference Database Access URL	AccessURL out of date	Verification of external links
R44	Review Status	ReviewStatus setting inconsistent with reviewDate / last structureModification	Verification of ReviewStatus accuracy
R45	SimpleEntity Has Species	SimpleEntity Has Species	Data internal consistency
R46	Species In Preceding Event	Inconsistent Species In Preceding Event	Data internal consistency
R47	Stable Identifier	No longer needed (?)	Data internal consistency
R48	Two Attributes Refer To Same Instance	Two attributes of one instance are identical but shouldn't be (eg species/relatedSp	Data internal consistency
R49	Two Attributes Refer To Same Instance	Two attributes of one instance are identical but shouldn't be (eg species/relatedSp	Data internal consistency
R50	Values Not Unique	Two attributes of one instance are identical but shouldn't be (eg species/relatedSp	Data internal consistency
CL1	Complex Components Species Mismatch	Complex Components Species Mismatch	Data internal consistency
CL2	Deleted Objects In Diagram	Deleted Objects In Diagram	Consistency between pathway diagrams and database content
CL3	Diagram EWAS Modification Mismatch	Diagram EWAS Modification Mismatch	Consistency between pathway diagrams and database content
CL4	Diagram Subpathway Undiagrammed Reactio	Diagram Subpathway Undiagrammed Reactions	Consistency between pathway diagrams and database content
CL5	Diagram Unrepresented Reactions	Diagram Unrepresented Reactions	Consistency between pathway diagrams and database content
CL6	Diagram Unsynchronized Reactions	Diagram Unsynchronized Reactions	Consistency between pathway diagrams and database content
CL7	Diagram With Missing Reactions	Diagram With Missing Reactions	Consistency between pathway diagrams and database content
CL8	Diagram With Unconnected Entities	Diagram With Unconnected Entities	Consistency between pathway diagrams and database content
CL9	Disease Entity Inconsistent	Verify	Data internal consistency
CL10	EntitySet Members Species Mismatch	EntitySet Members Species Mismatch	Data internal consistency
CL11	Extra Compartments In Entity Set Or Member	Extra Compartments In Entity Set Or Members	Data internal consistency
CL12	Instances Without StableIdentifier	Instances Without StableIdentifier	Data internal consistency
CL13	Mandatory Attributes	Mandatory Attributes	Data internal consistency
CL14	Normal Entity Inconsistent	Verify	Data internal consistency
CL15	Pathway Without Diagram	Pathway Without Diagram	Consistency between pathway diagrams and database content
CL16	Reaction Input Output Imbalance	Reaction Input Output Imbalance	Data internal consistency

remove from public list: add to JIRA too remove from pipeline
add to weekly QA: JIRA

Resolve duplicate: Keen R48 add this to Joomla

Cell: A122
Comment: check definition
:Lisa Matthews

Cell: A126
Comment: check
:Lisa Matthews

Cell: A132
Comment: deleted in public
:Lisa Matthews

Cell: A133
Comment: check
:Lisa Matthews

Cell: A134
Comment: Add to weeklyQA
:Lisa Matthews

Cell: A137
Comment: deleted in public
:Lisa Matthews

Cell: A139
Comment: deleted in public
:Lisa Matthews

Code	Name	Description	Category	
DT101	DiagramEmpty	Detects diagrams without any nodes inside them.	Consistency between pathway diagrams and database content	
DT102	MissingSchemaClass	Detects diagrams containing entities without a SchemaClass.	Consistency between pathway diagrams and database content	
DT103	ExtraParticipantInDiagram	Participants seen in a diagram that shouldn't be there. Mainly the reason is because	Consistency between pathway diagrams and database content	
DT104	DuplicatedReactionParticipants	Detects diagram reactions with duplicate inputs, outputs, catalysts, activators, inh	Consistency between pathway diagrams and database content	
DT111	UnrecognisedRenderableClass	A diagram glyph has an unrecognised or unsupported RenderableClass (Please	Consistency between pathway diagrams and database content	
DT112	UnidentifiedCompartments	A diagram contains an unidentified compartment (report to the developers list)	Consistency between pathway diagrams and database content	
DT401	SubpathwaysWithoutParticipants	Detects subpathways that include no diagram participants.	Consistency between pathway diagrams and database content	
DT701	ReactionParticipantsMismatch	Detects mismatches between the participants of a reaction in a diagram and the	Consistency between pathway diagrams and database content	
DT702	DiagramMissingReaction	Detects diagrams with missing reactions compared to how they are annotated in	Consistency between pathway diagrams and database content	
DT703	ExtraReactionInDiagram	Reactions seen in a diagram that shouldn't be there. Mainly the reason is because	Consistency between pathway diagrams and database content	
DT106	SchemaClassMismatch	SchemaClass of a diagram entity does not match the one stored in the database	Consistency between pathway diagrams and database content	
DT109	OverlappingReactions	Detects pairs of reactions which main shapes physically overlap in the diagram.	Readability and navigability of ELV pathway diagrams	
DT115	NodeAttachmentMismatch	Entities in the diagram that contain different node attachments to those annotated	Consistency between pathway diagrams and database content	
DT116	StoichiometryMismatch	Reaction participants where stoichiometry is different to the annotated in the data	Consistency between pathway diagrams and database content	
DT117	ParticipantWrongRole	Participants whose role is wrongly displayed in the diagram.	Consistency between pathway diagrams and database content	
DT105	RenderableClassMismatch	Detects diagrams containing entities with a mismatch in their annotated Rendera	Consistency between pathway diagrams and database content	
DT107	WronglyPlacedReactionShape	Detects cases where the reaction shapes (Rectangle for transition, Circle for bind	Consistency between pathway diagrams and database content	
DT113	ReactionShapeMismatch	Detects reaction categories that differ from the expected ones based on the ann	Consistency between pathway diagrams and database content	
DT904	PhysicalEntityWrongTranslationalModification	PhysicalEntity instances with a potentially wrongly annotated translational modifi	Consistency between pathway diagrams and database content	
DT108	IsolatedGlyphs	Iterates over all Edges and Links and identifies the isolated glyphs. The method	Readability and navigability of ELV pathway diagrams/Consistency between pathway diagrams and database content	
DT110	OverlappingEntities	Detects pairs of entities that physically overlap in the diagram.	Readability and navigability of ELV pathway diagrams	
DT114	ReplacedNodeAttachmentLabel	Fixes the node attachments labels based on the database annotation and report	Consistency between pathway diagrams and database content	
GT003	DatabaseIdentifierWithoutIdentifier	DatabaseIdentifier class instances where the identifier slot is empty	Data internal consistency	
GT005	PathwaysWithoutEvents	Pathway class instances where the hasEvent slot is empty	Data internal consistency	
GT006	EwasWithoutReferenceEntity	EWAS class instances where the referenceEntity slot is empty	Data internal consistency	
GT007	EntitiesWithoutStd	Event and PhysicalEntity class instances where stableIdentifier slot is empty	Data internal consistency	
GT009	ComplexWithoutComponents	Complexes instances where the hasComponent slot is empty	Data internal consistency	
GT012	SimpleEntityWithoutReferenceEntity	SimpleEntity class instances where referenceEntity slot is empty	Data internal consistency	
GT013	ReferenceEntityWithoutIdentifier	ReferenceEntity class instances where the identifier slot is empty	Data internal consistency	
GT022	PhysicalEntityWithoutCompartment	PhysicalEntity class instances where the compartment slot is empty	Data internal consistency	
GT101	CompartmentWithCyclicSurroundedByAndIn	Compartments with cyclic relationships in the surroundedBy, componentOf and/o	Data internal consistency	
GT102	FiguresURLInconsistence	Figures (ehld/static) with potential mismatch or missing extension	Data internal consistency	
GT004	EntriesWithoutDisplayName	The instance's displayName slot is empty	Data internal consistency	
GT010	EntitySetWithoutMemberOrCandidate	EntitySet instances where both hasMember and hasCandidate slots are empty	Data internal consistency	
GT011	PolymerWithoutRepeatedUnit	Polymer class instances where the repeatedUnit slot is empty	Data internal consistency	
GT015	CatalystActivityWithoutPhysicalEntity	CatalystActivity class instances where the physicalEntity slot is empty	Data internal consistency	
GT016	CatalystActivityWithoutActivity	CatalystActivity class instances where the activity slot is empty	Data internal consistency	
GT017	NOT_FailedReactionsWithoutOutputs	ReactionLikeEvent class instance (excluding FailedReaction class instances) wh	Data internal consistency	
GT019	PublicationsWithoutAuthor	Publication class instances where the author slot is empty	Data internal consistency	
GT021	RegulationsWithoutRegulatedEntityOrRegula	Regulation class instances where the regulatedEntity or the regulator slots is em	Data internal consistency	
GT024	ReferenceDatabaseWithoutUrls	ReferenceDatabase class instances where accessUri slot or url slot are empty (e	Data internal consistency	
GT028	HasMemberAndHasCandidatePointToSame	CandidateSet class instances where at least one instance in the hasCandidate s	Data internal consistency	
GT029	ReactionsLikeEventWithoutInput	ReactionLikeEvent instances where the input slot is empty	Data internal consistency	
GT036	EventsWithoutCompartment	Event class instances where the compartment slot is empty	Data internal consistency	
GT037	LiteratureReferenceRelationshipDuplication	DatabaseObject class instances where the literatureReference slot contains dupl	Data internal consistency	
GT038	HasModifiedResidueRelationshipDuplication	DatabaseObject class instances where the hasModifiedResidue slot contains dup	Data internal consistency	
GT040	HasMemberRelationshipDuplication	DatabaseObject class instances where the hasMember slot contains duplicated e	Data internal consistency	
GT043	HasCandidateRelationshipDuplication	CandidateSet class instances where the hasCandidate slot contains duplicated e	Data internal consistency	
GT044	HasEventRelationshipDuplication	Event class instances where the hasEvent slot contains duplicated entries	Data internal consistency	
GT050	DuplicatedLiteratureReferences	Different instances of the LiteratureReference class with the same PubMed ident	Data internal consistency	
GT055	DuplicatedCuratedComplexes	Two different instances of the class Complex that have the same instances in the	Data internal consistency	
GT057	ComplexesWithOnlyOneComponent	Complex class instances with only one entry in the hasComponent slot	Data internal consistency	
GT058	ComplexesWhereCompartmentDoesNotMatc	Complexes where the compartment does not matc wWith any of the participants	Data internal consistency	
GT059	DuplicatedCandidateSets	Two different instances of the class CandidateSet that have the same instances i	Data internal consistency	
GT062	DuplicatedReferenceEntities	Two different instances of the class ReferenceEntity that have the same content	Data internal consistency	
GT064	DuplicatedEntitySets	Two different instances of the class EntitySet that have the same content	Data internal consistency	
GT070	ReactionsWithoutRegulatorWithMoreCompa	Reactions without regulator with more compartments than its participants	Data internal consistency	

GT071	ReactionsWithOnlyOneInputAndOutputWhere	Reactions with only one input and output where schemaClass do not match	Data internal consistency	
GT090	CatalystActivityCompartmentDoesNotMatchF	CatalystActivity class instances where the compartment slot does not match with	Data internal consistency	
GT091	ReactionsWithoutLiteratureReference	Reactions with no LiteratureReference class instances in any of the possible loca	Missing required experimental evidence, Data internal consistency	
GT092	PotentialTranslocationReactionChangesParti	Potential translocation Reaction changes participants schemaClass	Data internal consistency	
GT100	DiseasePathwayWithoutNormalPathwayOrN	Disease Pathway instances without normalPathway or NormalPathway has no di	Data internal consistency	
GT001	DatabaseObjectsWithSelfLoops	The instance contains itself in any of the slots	Data internal consistency	
GT020	OpenSetsWithoutReferenceEntity	OpenSet class instances where the referenceEntity slot is empty	Data internal consistency	
GT023	DatabaselfIdentifierWithoutReferenceDatabas	DatabaselfIdentifier class instances where the referenceDatabase slot is empty	Data internal consistency	
GT025	EntriesWithCyclicInferredToRelations	When DatabaseObjec class instance (A) is used to infer another instance (B), if A	Data internal consistency	
GT026	EventsWithCyclicPrecedingEvents	When an Event class instance (A) contains another instance (B) in the preceding	Data internal consistency	
GT027	EntriesWithOtherCyclicRelations	Same concept as the two above but checking for other slots (and not focusing on	Data internal consistency	
GT030	PhysicalEntitiesWithMoreThanOneCompartin	PhysicalEntity class instances where the compartment slot contains more than of	Data internal consistency	
GT031	CatalystActivityWherePhysicalEntityAndActiv	CatalystActivity class instances where the activity and activeUnit slots point to the	Data internal consistency	
GT032	PrecedingEventOrReverseReactionOrHasEv	PrecedingEvent or ReverseReaction or HasEvent point to same Event instance	Data internal consistency	
GT035	CrossReferenceRelationshipDuplication	DatabaseObject class instances where the crossReference slot contains duplicat	Data internal consistency	
GT039	PrecedingEventRelationshipDuplication	DatabaseObject class instances where the precedingEvent slot contains duplicat	Data internal consistency	
GT041	SummationRelationshipDuplication	DatabaseObject class instances where the summation slot contains duplicated e	Data internal consistency	
GT042	PsiModRelationshipDuplication	DatabaseObject class instances where the psiMod slot contains duplicated entrie	Data internal consistency	
GT047	OrphanEvents	Events that cannot be reached through the events hierarchy	Data internal consistency	
GT051	PersonsWithSameORCIDAndProject	Different instances of the Person class with the same ORCID identifier (and proje	Contributor attribution check	
GT053	PrecedingEventOutputsNotUsedInReaction	ReactionLikeEvent (A) annotated as preceding of another one (B) where none of	Data internal consistency	
GT061	EntitySetsWithOnlyOneMember	EntitySet class instances (excluding CandidateSet and OpenSet) that has only of	Data internal consistency	
GT065	EntitySetsWithRepeatedMembers	EntitySet class instances (excluding CandidateSet) where the same PhysicalEnti	Data internal consistency	
GT002	PersonWithoutProperName	Surname is empty or the Firstname and the Initial are also empty	Data internal consistency	
GT014	InstanceEditWithoutAuthor	InstanceEdit class instances where the author slot is empty	Data internal consistency	
GT018	DatabaseObjectsWithoutCreated	DatabaseObject class instances (excluding InstanceEdit, DatabaselfIdentifier, Tax	Data internal consistency	
GT033	OtherRelationsThatPointToTheSameEntry	Other relations that point to the same entry	Data internal consistency	
GT034	ModifiedRelationshipDuplication	DatabaseObject class instances where the modified slot contains duplicated entr	Data internal consistency	
GT045	OtherRelationshipDuplication	DatabaseObject class instances with duplicated instances in a multivalued slot (d	Data internal consistency	
GT048	InstanceEditCreatesInstanceEdit	InstanceEdit class instances that are created by other InstanceEdit class instanc	Data internal consistency	
GT049	InstanceEditModifiesInstanceEdit	InstanceEdit class instances that are modified by other InstanceEdit class instanc	Data internal consistency	
GT052	PersonsWithSameNameAndProject	Different instances of the Person class with the same name and project	Data internal consistency	
R1	Attribute Has Multiple Values	Attribute Has Multiple Values where should only have one	Data internal consistency	
R2	Attribute Has Only One Value	Attribute Has Only One Value where should have >1	Data internal consistency	
R3	Attribute Value Duplication	Attribute Value Duplication	Data internal consistency	
R4	Attribute Value Missing	CatalystActivity PhysicalEntity ActivityUnit Refers To Same Complex	Data internal consistency	
R5	CatalystActivity PhysicalEntity ActivityUnit Re	ActivityUnit only used if more specific than the P.E	Data internal consistency	
R6	Chimerism Reference Constraint Violations	Verifies Instances with IsChimeric attribute have other expected attributes	Data internal consistency	
R7	Complexes That Should Have Homo sapiens	Complexes That Should Have Homo sapiens Species	Data internal consistency	
R8	CoV-2 Entities With CoV-1 Species Or Displa	CoV-2 Entities With CoV-1 Species Or DisplayName	Data internal consistency	
R9	CoV-2 Infection Pathway Events With Summ	CoV-2 Infection Pathway Events With unmodified Summation And Literature Ref	Verifying manual review of inferred/new/revised data	
R10	DatabaseObject With Self Loop	Class instance should not have instances that referring to self	Readability and navigability of ELV pathway diagrams	
R11	Diagram Compartment Label Missing	Diagram Compartment Label Missing	Readability and navigability of ELV pathway diagrams	
R12	Diagram Disease Color	Disease entities in diagram must be updated	Consistency between pathway diagrams and database content	
R13	Diagram Drug Color	Drug entities in diagram must be updated	Consistency between pathway diagrams and database content	
R14	Diagram Duplicate Reaction Participants	Duplicate Reaction Participants in a diagram	Consistency between pathway diagrams and database content	
R15	Diagram Empty	Diagram Empty	Consistency between pathway diagrams and database content	
R16	Diagram Extra ReactionLikeEvents	Extra ReactionLikeEvents in diagram that are not in the hierarchy in database	Consistency between pathway diagrams and database content	
R17	Diagram Overlapping Entities	Diagram with Overlapping Reactions	Consistency between pathway diagrams and database content	
R18	Diagram Overlapping Reactions	Diagram Reactions With Participant Overlapping Reaction Hub	Readability and navigability of ELV pathway diagrams	
R19	Diagram Reactions With Participant Overlap	Diagram has reactions with participant overlapping the hub	Readability and navigability of ELV pathway diagrams	
R21	EHL Diagram Subpathway Change	Pathways associated with an EHL diagram that have had a hierarchy structure	Consistency between pathway diagrams and database content	
R22	Entities With CoV Species Without Correspo	Entities With CoV Species without corresponding disease tag	Data internal consistency	
R23	Events With Electronic Evidence Types	Events that have been inferred from electronic inference during orthoprojection	Data internal consistency	
R24	FailedReaction Has Output	FailedReaction with an Output	Data internal consistency	
R25	Human Reactions Without Disease And Hav	Human reactions without a disease attribute but having Non Human PhysicalEnt	Data internal consistency	
R26	Inferred Modified Residues	Modified residues in slice database that still contain [INFERRED] tag and need to	Data internal consistency	
R27	InferredFrom Used In Other Attribute	InferredFrom Used In Other Attribute	Verifying manual review of inferred/new/revised data	
R28	Instance Duplication	Instance Duplication	Data internal consistency	
R29	Missing Editorial Attribute	Missing Editorial Attribute	Verifying manual review of inferred/new/revised data	
R30	Multiple Attributes Cross Classes Missing Sin	Missing attributes in indicated classes	Missing required experimental evidence, Data internal consistency	
R31	Multiple Attributes Missing Simultaneously	Certain attributes for a given class should not be missing simultaneously, e.g if	Missing required experimental evidence, Data internal consistency	
R32	New Event Inconsistent	New event missing expected attributes	Data internal consistency	
R33	NonHuman Events Not Manually Inferred	Non-human events under Human hierarchy should have species=H.s. unless us	Data internal consistency	
R34	NonHuman Reactions Containing Human Ph	Non-human events NOT under Human hierarchy should not have entities with sp	Data internal consistency	
R35	One Hop Circular Reference	Attribute assignment in instance creates an inaccurate circular reference	Data internal consistency	
R36	Orcid Crossreference	ORCID inconsistency	Contributor attribution check	
R37	Orphan Events	Event is set as doRelease =True but not connected to the main event hierarchy	Data internal consistency	
R38	PhysicalEntity With More Than One Compart	PhysicalEntity With More Than One Compartment	Data internal consistency	

R39	PhysicalEntity Without Species Components	PhysicalEntity species is null but components species are not null	Data internal consistency	
R40	Reaction Single Input Output Schema Not Match	Molecule type is different in the input vs. output	Data internal consistency	
R41	ReactionlikeEvent Not Failed With Normal With	ReactionlikeEvent Not Failed but has NormalEvent and not Disease	Data internal consistency	
R43	Reference Database Access URL	AccessURL out of date	Verification of external links	
R44	Review Status	ReviewStatus setting inconsistent with reviewDate / last structurelModification	Verification of ReviewStatus accuracy	
R45	SimpleEntity Has Species	SimpleEntity Has Species	Data internal consistency	
R46	Species In Preceding Event	Inconsistent Species In Preceding Event	Data internal consistency	
R48	Two Attributes Refer To Same Instance	Two attributes of one instance are identical but shouldn't be (eg species/relatedSpecies)	Data internal consistency	
R50	Values Not Unique	Two attributes of one instance are identical but shouldn't be (eg species/relatedSpecies)	Data internal consistency	
CL1	Complex Components Species Mismatch	Complex Components Species Mismatch	Data internal consistency	
CL2	Deleted Objects In Diagram	Deleted Objects In Diagram	Consistency between pathway diagrams and database content	
CL3	Diagram EWAS Modification Mismatch	Diagram EWAS Modification Mismatch	Consistency between pathway diagrams and database content	
CL4	Diagram Subpathway Undiagrammed Reactions	Diagram Subpathway Undiagrammed Reactions	Consistency between pathway diagrams and database content	
CL5	Diagram Unrepresented Reactions	Diagram Unrepresented Reactions	Consistency between pathway diagrams and database content	
CL6	Diagram Unsynchronized Reactions	Diagram Unsynchronized Reactions	Consistency between pathway diagrams and database content	
CL7	Diagram With Missing Reactions	Diagram With Missing Reactions	Consistency between pathway diagrams and database content	
CL8	Diagram With Unconnected Entities	Diagram With Unconnected Entities	Consistency between pathway diagrams and database content	
CL9	Disease Entity Inconsistent	Disease Entity in the EntityFunctionalStatus of a disease reaction is not a participant	Data internal consistency	
CL10	EntitySet Members Species Mismatch	EntitySet Members Species Mismatch	Data internal consistency	
CL11	Extra Compartments In Entity Set Or Members	Extra Compartments In Entity Set Or Members	Data internal consistency	
CL12	Instances Without StableIdentifier	Instances Without StableIdentifier	Data internal consistency	
CL13	Mandatory Attributes	Mandatory Attributes	Data internal consistency	
CL14	Normal Entity Inconsistent	NormalEntity in entityFunctionalStatus of a disease reaction is not a participant	Data internal consistency	
CL15	Pathway Without Diagram	Pathway Without Diagram	Consistency between pathway diagrams and database content	
CL16	Reaction Input Output Imbalance	Reaction Input Output Imbalance	Data internal consistency	

Appendix R7: Updating Documentation on S3, website and Zenodo with each release

Link to [background information](#) describing rationale for this

This step happens after the release is complete (to allow for the capture of all changes to the Release SOP for that release). Do after release is fully live and release P2P is done.

Up to 4 docs need to be deposited into S3, updated on website and deposited in Zenodo
Reactome Documentation with each release:

- [Release SOP](#), plus [Appendices](#) – with every release, as generally something changes each time. The docs are in [Team Drive/Documentation/Release SOP](#). Note that we are not currently implementing any “track changes” note-taking on this doc aside from what is provided inherently by Google Docs revision history.
- [Curator Guide](#) plus [Appendices](#). The docs are in [Team/Documentation/Curation/CuratorGuide](#). See [section below](#) for how to track changes and update the Curator Guide docs.
- User Guide –The doc is [here](#), in [Team/Documentation/Outreach/Website/UserGuide](#)-infrequently updated.
- Data Model Glossary – doc is [here](#), in [Team/Documentation/Curation/Data Model Glossary](#)- infrequently updated.

For the Release SOP and Appendices and Curator Guide and Appendices:

- for each release, add any new appendices to the master list of appendices, providing a link to the new appendix.
 - curation appendix list [here](#).
 - Curator Guide appendices are numbered C (for CuratorGuide) 1, 2, etc. Ideally, name the doc with the appropriate appendix number (ie “C16: Rhea:GO alignment”) at the top of the first page of the doc so that when it is PDF’d into the enormous Curator Guide for public upload there is some kind of demarcation between one appendix and the next.
 - release SOP appendix list [here](#).
 - Release SOP appendices are numbered R (for release SOP) 1, 2, etc. Again, ideal to label the doc with R## for PDF’ing purposes
- update footer on any docs that have been updated
- download all files, generally as .docx unless an Excel in which case PDF. For downloaded .docx files, turn off track changes to get rid of marginal comments before PDF’ing
- save as PDFs
- make two new enormous PDFs, CuratorGuideAndAppendices_VXX and ReleaseSOPandAppendices_VXX, using ie Adobe Acrobat to combine the main doc with all of its appendices into two single PDFs. **Note that for S3 and Joomla-linking purposes, the names of these PDFs can’t have special characters or spaces.**
 - put combined CuratorGuideAndAppendices PDF into a [version folder here](#) in Team Drive
 - put combined ReleaseSOPandAppendices PDF into a [version folder here](#) in Team Drive

In most cases, User Guide and Data Model Glossary will not have any changes since last release, and the old PDF's can be used. If any changes have been made to these two documents, a new PDF will need to be generated.

Once all four PDFs are ready, they are put into S3, the website is updated with the S3 link for the ReleaseSOP and CuratorGuide, and the 4 docs are also put into Zenodo. For S3, we append _VXX to PDF name but remove the _VXX from name for Zenodo.

S3:

Sign in to [S3 on AWS](#)

Click s3

Click download.reactome.org

Click Documentation

Click Upload.

- Upload (drag and drop) only the docs that have changed; if something is unchanged from previous release, don't need to reupload.
- Make sure no special characters or spaces in PDF names. If you need to rename because of inappropriately formatted document name, select the item and the select "rename" under the Action drop down at the top.

Once docs are uploaded, copy the S3 URI url for the CuratorGuideandAppendices and the ReleaseSOPandAppendices to add to Joomla. Don't take the s3:// part, just from download on: [s3://download.reactome.org/documentation/CuratorGuideAndAppendices_V90.pdf](https://download.reactome.org/documentation/CuratorGuideAndAppendices_V90.pdf) - but add in <https://>

Joomla:

Sign into Joomla at release.reactome.org

- CuratorGuideandAppendices link has to be added to the article under Docs/CuratorGuide. Final link format https://download.reactome.org/documentation/CuratorGuideAndAppendices_VXX.pdf
- ReleaseSOPandAppendices link has to be added to the article under Docs/ReleaseDocumentation. Final link format: https://download.reactome.org/documentation/ReleaseSOPandAppendices_VXX.pdf

Zenodo:

For Zenodo, we upload each doc for every release, regardless of whether it has changed since the previous release or not. For Zenodo files, remove the _VXX from the PDF names.

Sign into [Zenodo](#)

Click 'Communities' from top banner, search for Reactome.

Select Reactome documentation

Click "new version"

Click "upload files", select files (should be 4 PDFs)

Scroll down in upload window to add publication date (ie Sept 2024) and version number (just ##, no word "version")

Click upload

Preview to make sure things look right

"back to edit"

Publish.

Versioning/Tracking changes in CuratorGuide and associated appendices:

Updating:

Any time a change is made to any of the documents (main or appendices), a brief description of the change is captured in a footer on the bottom of the last page of the internal document, marked with the next release version and the date the change was made (Month Year).

ie V90_Sept 2024

- moved from DevWiki
- examples updated with images
- grey out references to gk_central and live site remote attribute search

Any time a document (main or appendix) is updated, a new PDF must be created of that document and a new package of (main + appendices) made available for the next release.

Documents that are unchanged from the previous release should not be updated and the version number should not be incremented with successive releases. The current, existing version of the doc will get incorporated into the updated release PDF.

The version number of the main doc should always be in sync with the version number of the most recently updated appendix.

Examples:

For Version 90 release, all curation documents (Guide + 15 appendices) are PDF'd with a _V90.

If between release 90 and release 91, no changes to any curation documentation are made:

- the V90 documents are left unchanged on the website for V91 - no action needed

If between release 90 and release 91 a change is made to the Curator Guide, but no changes are made to any of the appendices

- a footer with V91_(month, year of *date of change*) is added to the active, internal use Curator guide describing briefly what has changed
ie V91_Dec 2024
 - Updated description of approved drug usage
- Other unchanged appendices are left as V90
- New zipped PDFs of all appendices are put in the appropriate folder (V91 final curation PDFs) and put in S3
- Link on website is updated to new document, which (in this example) will now have Curator Guide_V91, Appendices 1-15 as _V90

If between release 90 and release 91 a change is made to ie curation appendix 5 (data model glossary):

- a footer with V91_(month, year of *date of change*) is added to the active, internal use document describing briefly what has changed

ie V91_Dec 2024

- added new data class x
- the footer on the main curator guide is updated to reflect the change (“V91_Dec 2024: update to appendix 5”)
- Other unchanged appendices are left as V90
- New zipped PDFs of all appendices are put in the appropriate folder (V91 final curation PDFs) and put in S3
- Link on website is updated to new document, which (in this example) will now have Curator Guide_V91, Appendices 1-4 and 6-15 as _V90, and Appendix 5 as _V91.