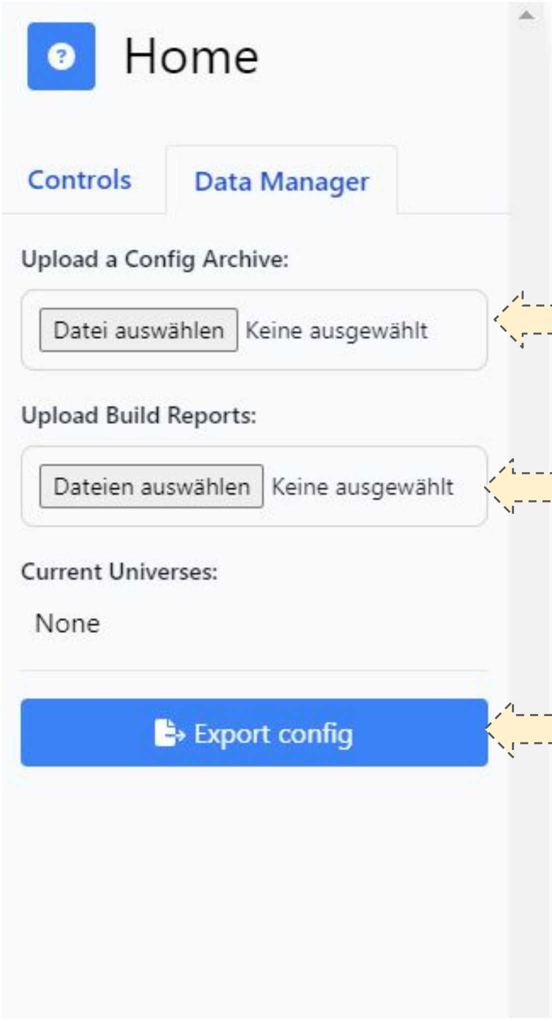
Observatory Usage

A Project By the Software Architecture Group’s Master Project
WS 22/23

General



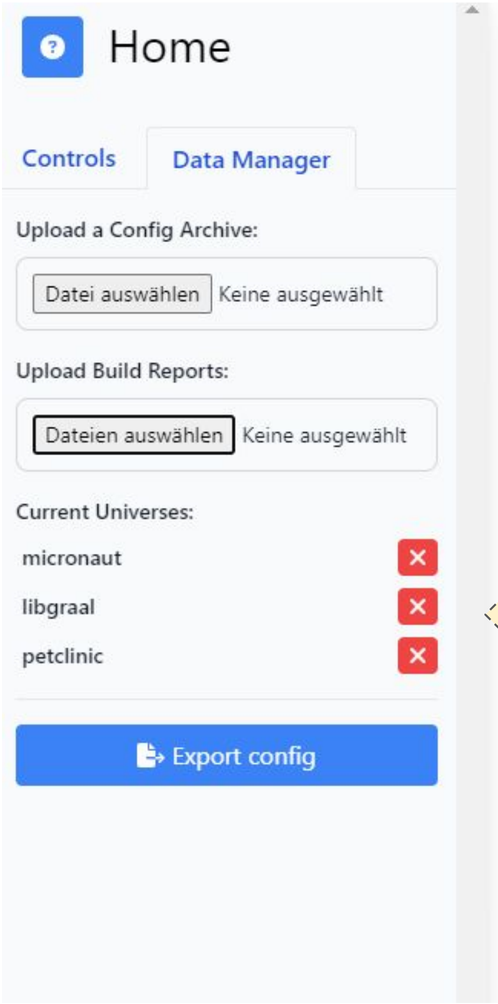
Current Universes:

None

Import a previously exported configuration of the observatory the recreate your state

Upload a Reachability or Causality Export of GraalVM Native Images (= Universes)

Export your current configurations, e.g. uploads, selectios, filters, … and share them with others



Upload a Config Archive:

Datei auswählen Keine ausgewählt

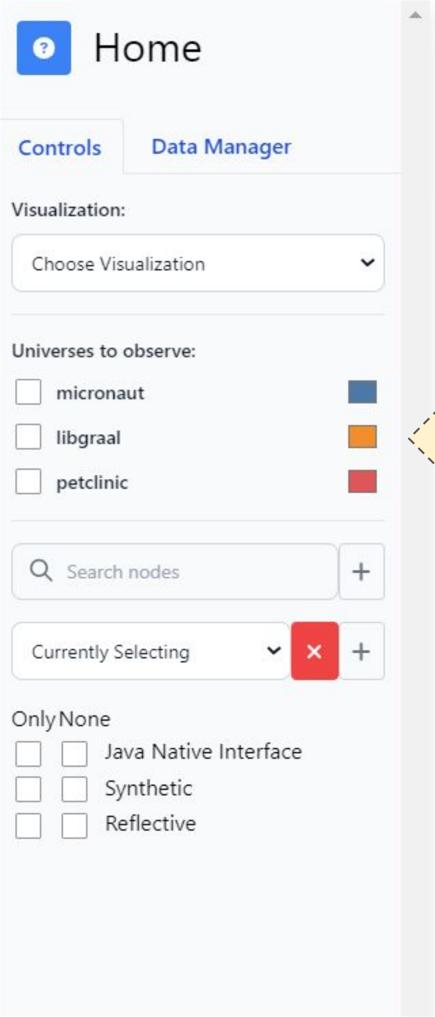
Upload Build Reports:

Dateien auswählen Keine ausgewählt

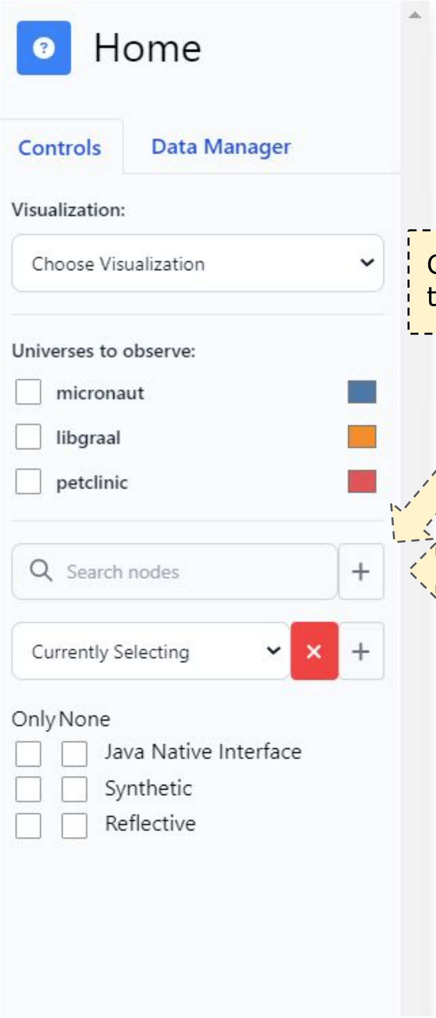
Current Universes:

- micronaut ✕
- libgraal ✕
- petclinic ✕

Your uploaded universes will appear here. Double click the text to rename them!

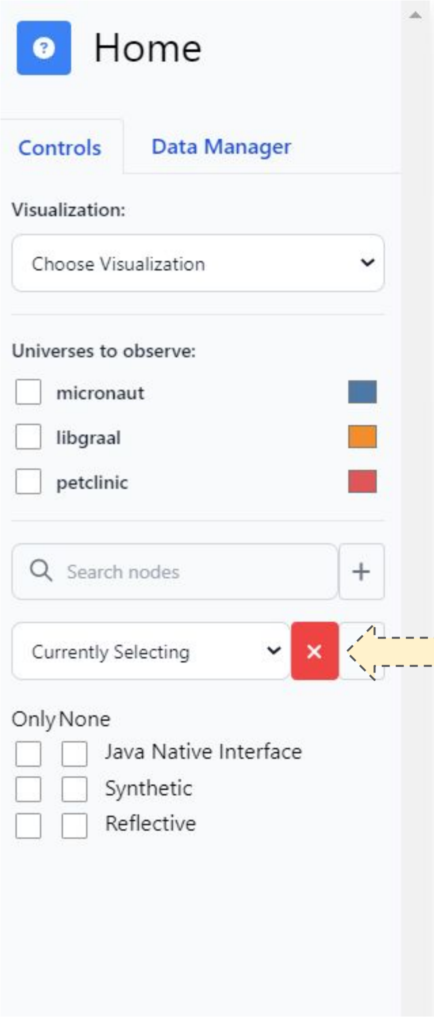


Set chosen universes visible in the visualization

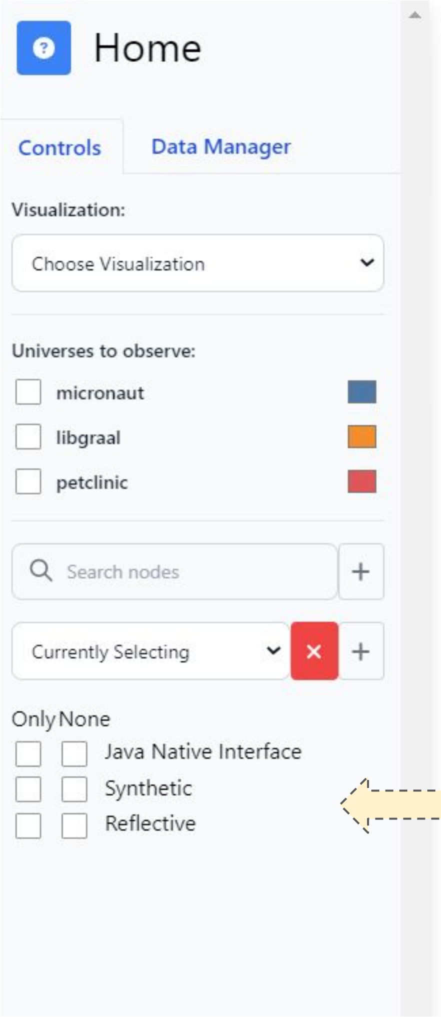


Click "+" to add the current term as a filter

Type in names of Modules, Packages, etc., to highlight every item containing the search term



Click to display all of your selected items. Usually by pressing (ctr +) left click on an item. The “x” will clear the selection



Home

Controls | Data Manager

Visualization:

Choose Visualization

Universes to observe:

- □ micronaut 🟦
- □ libgraal 🟧
- □ petclinic 🟥

Search nodes +

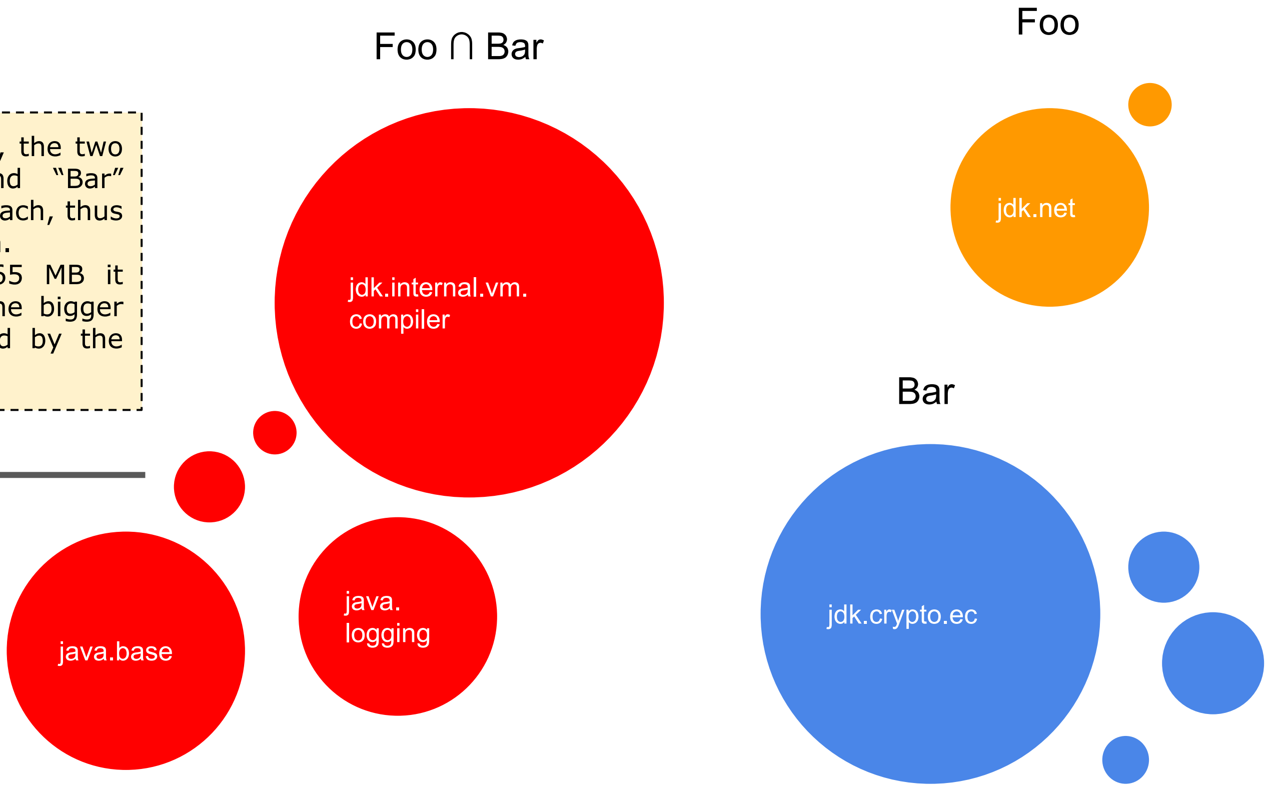
Currently Selecting × +

Only	None	
□	□	Java Native Interface
□	□	Synthetic
□	□	Reflective

Use the filters to display nodes only having a certain attribute. Or, all nodes **not** having this attribute.

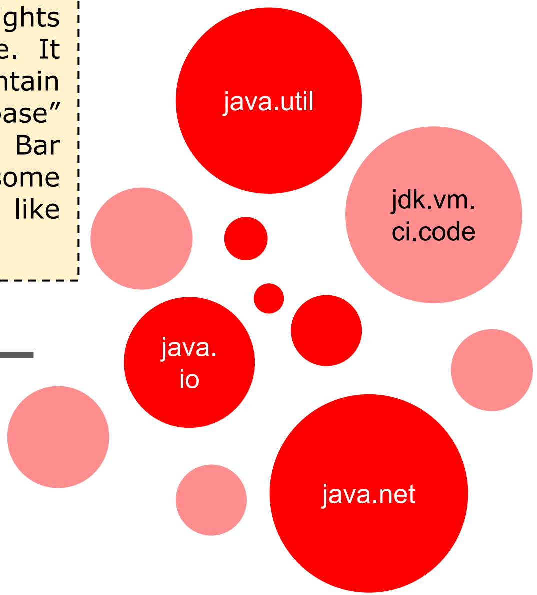
Venn Sets

On the module layer, the two universes “Foo” and “Bar” contain “java.base” each, thus it’s in the intersection.
With its size of 6.65 MB it belongs to one of the bigger modules, represented by the radius

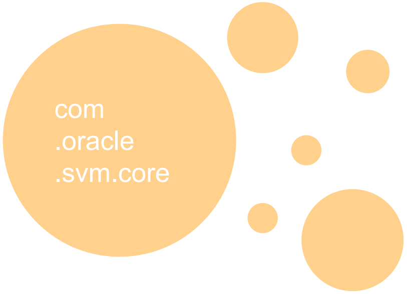


Using the slider to dive deeper, searching “java.base” highlights all packages in that module. It reveals that Foo doesn’t contain any packages of “java.base” which aren’t shared with Bar. Bar on the other hand has some exclusive packages like “java.base.javax.crypto”!

Foo ∩ Bar

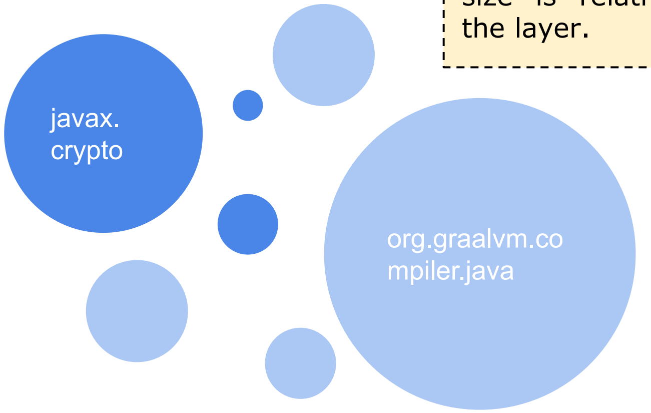


Foo



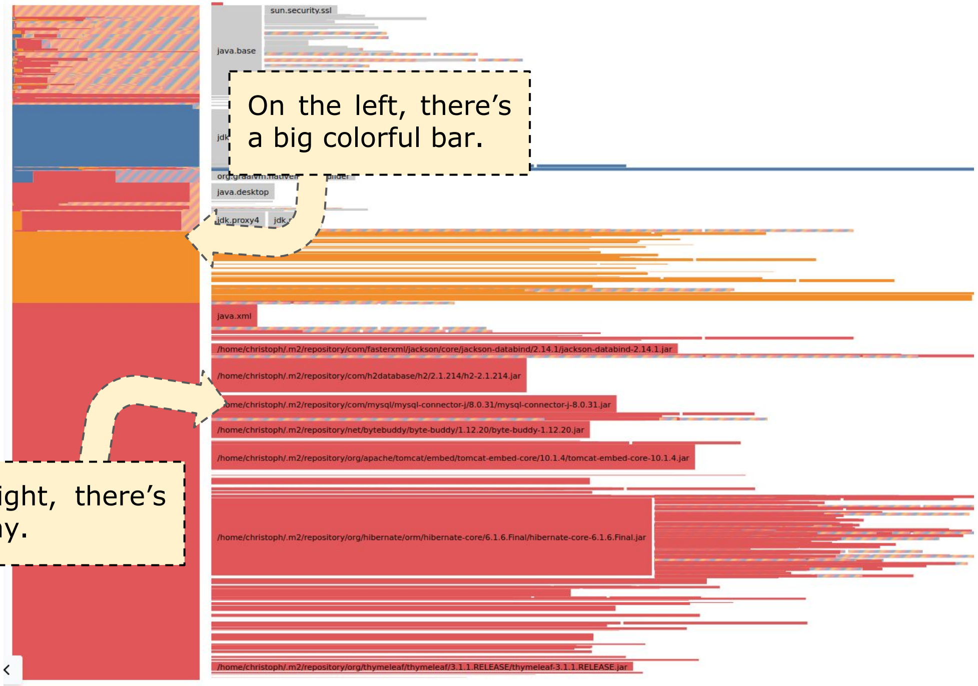
Even though “org.graalvm.compiler.java” is bigger than “java.base” one layer above, it’s only 400KB. Its size is relative to the layer.

Bar



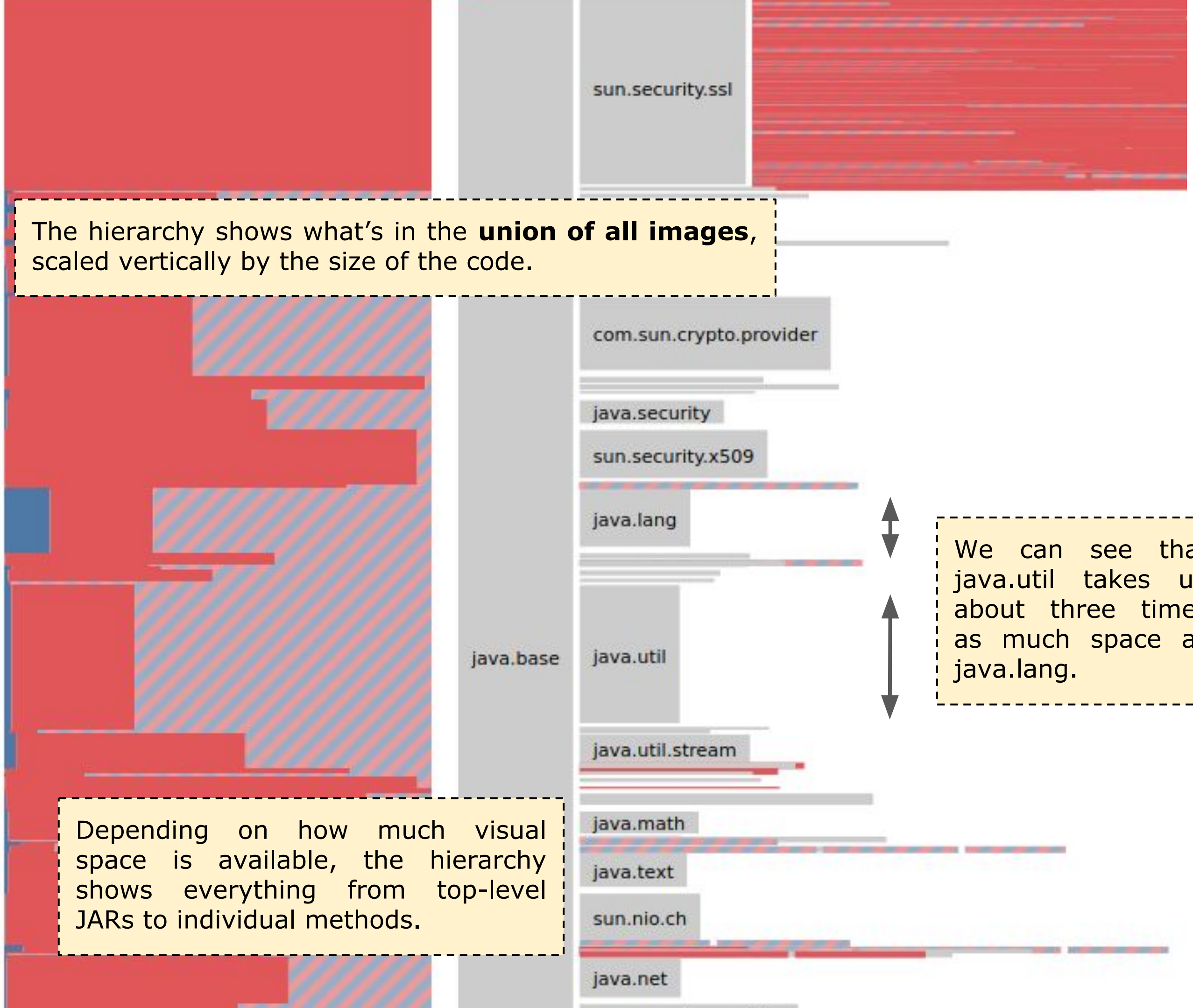
Tree Line

You can use the Tree Line to find out what images contain and which parts take up how much space. When diffing multiple images, each image has a color, which you can see in the configuration panel. The Tree Line below diffs three images (blue, orange, and red).


sun.security.ssl
java.base
jdk
org.graalvm.nativei
java.desktop
jdk.proxy4
jdk.
java.xml
/home/christoph/.m2/repository/com/fasterxml/jackson/core/jackson-databind/2.14.1/jackson-databind-2.14.1.jar
/home/christoph/.m2/repository/com/h2database/h2/2.1.214/h2-2.1.214.jar
ome/christoph/.m2/repository/com/mysql/mysql-connector-j/8.0.31/mysql-connector-j-8.0.31.jar
/home/christoph/.m2/repository/net/bytebuddy/byte-buddy/1.12.20/byte-buddy-1.12.20.jar
/home/christoph/.m2/repository/org/apache/tomcat/embed/tomcat-embed-core/10.1.4/tomcat-embed-core-10.1.4.jar
/home/christoph/.m2/repository/org/hibernate/orm/hibernate-core/6.1.6.Final/hibernate-core-6.1.6.Final.jar
/home/christoph/.m2/repository/org/thymeleaf/thymeleaf/3.1.1.RELEASE/thymeleaf-3.1.1.RELEASE.jar

On the left, there's a big colorful bar.

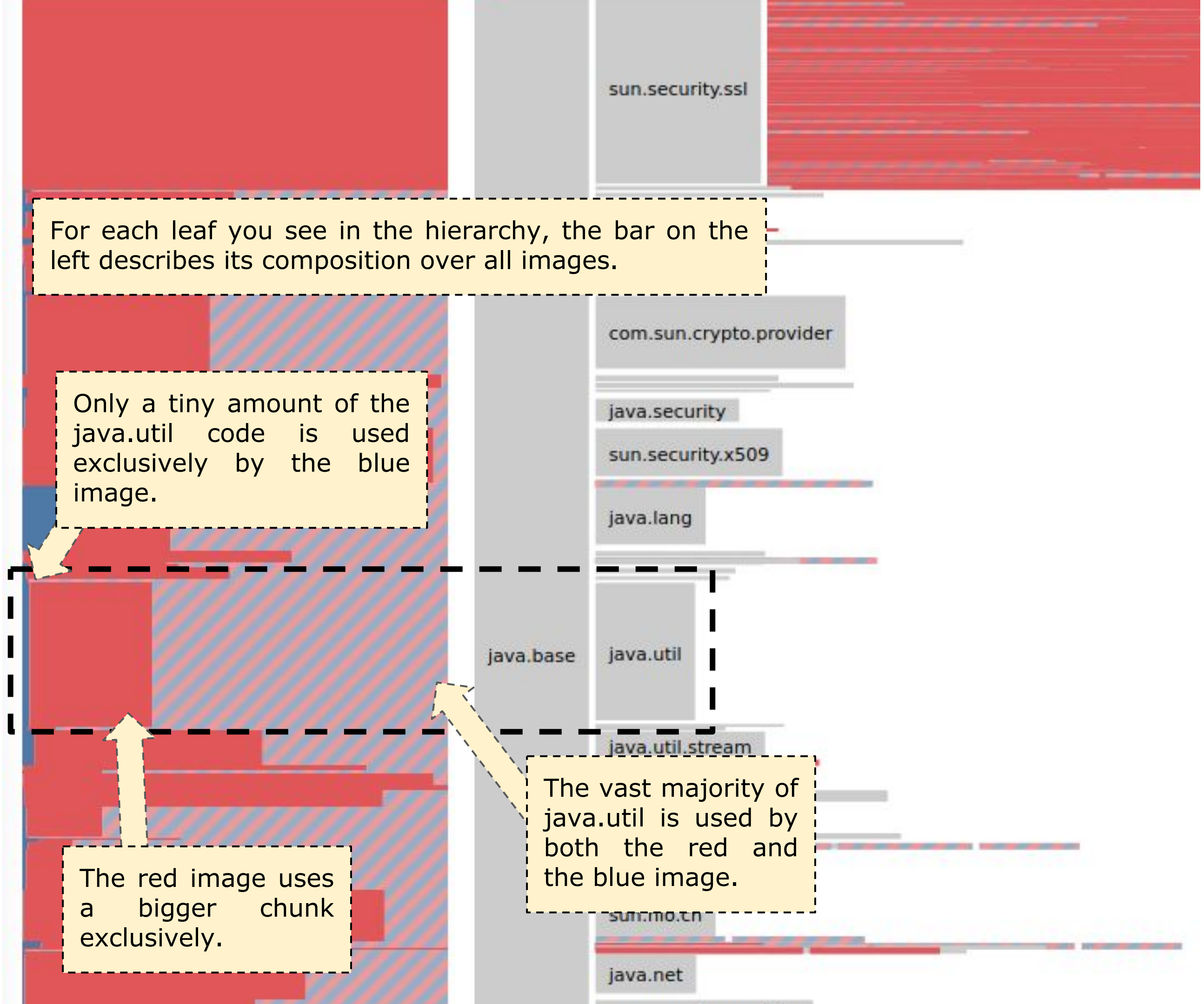
On the right, there's a hierarchy.



The hierarchy shows what’s in the **union of all images**, scaled vertically by the size of the code.

We can see that java.util takes up about three times as much space as java.lang.

Depending on how much visual space is available, the hierarchy shows everything from top-level JARs to individual methods.

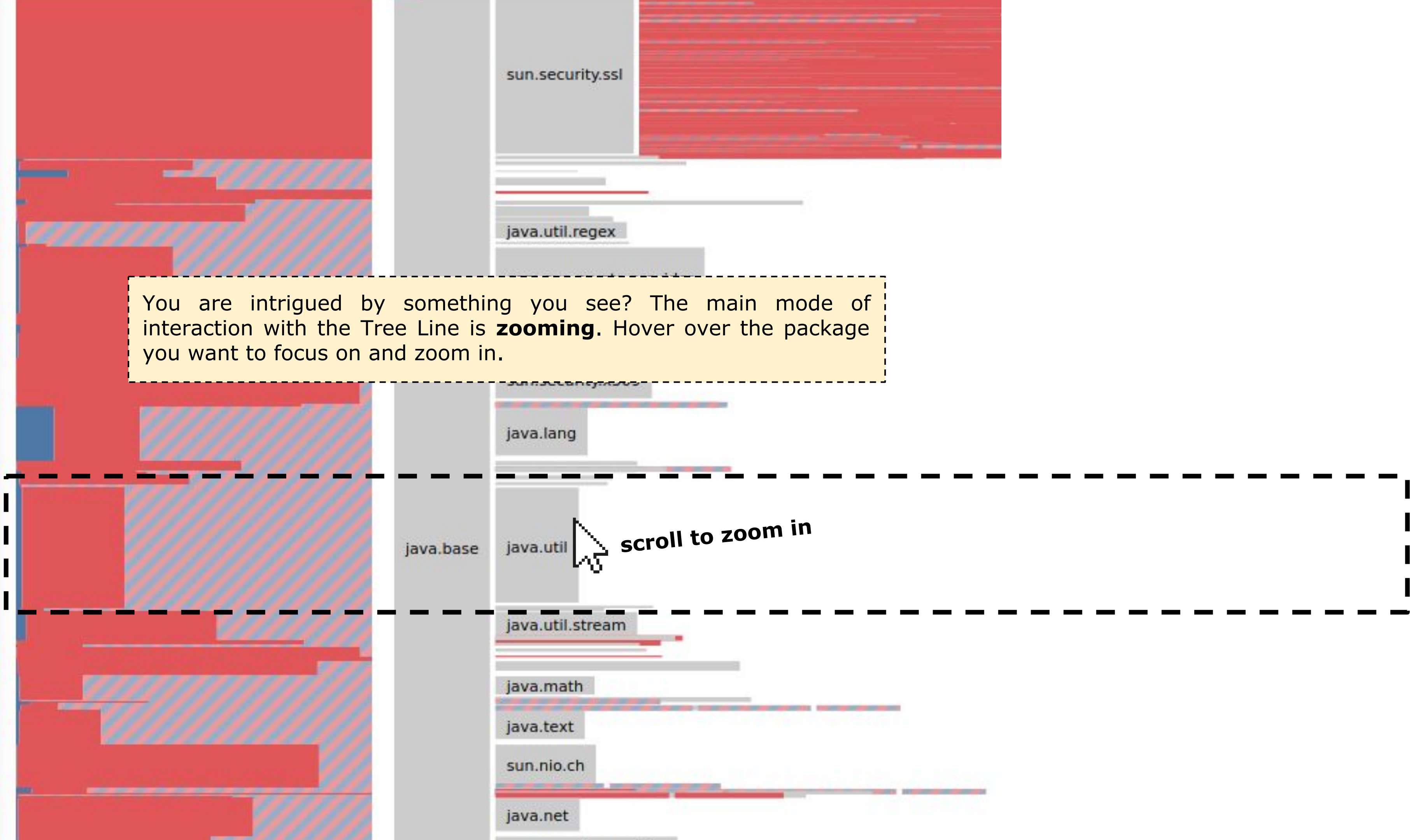


For each leaf you see in the hierarchy, the bar on the left describes its composition over all images.

Only a tiny amount of the java.util code is used exclusively by the blue image.

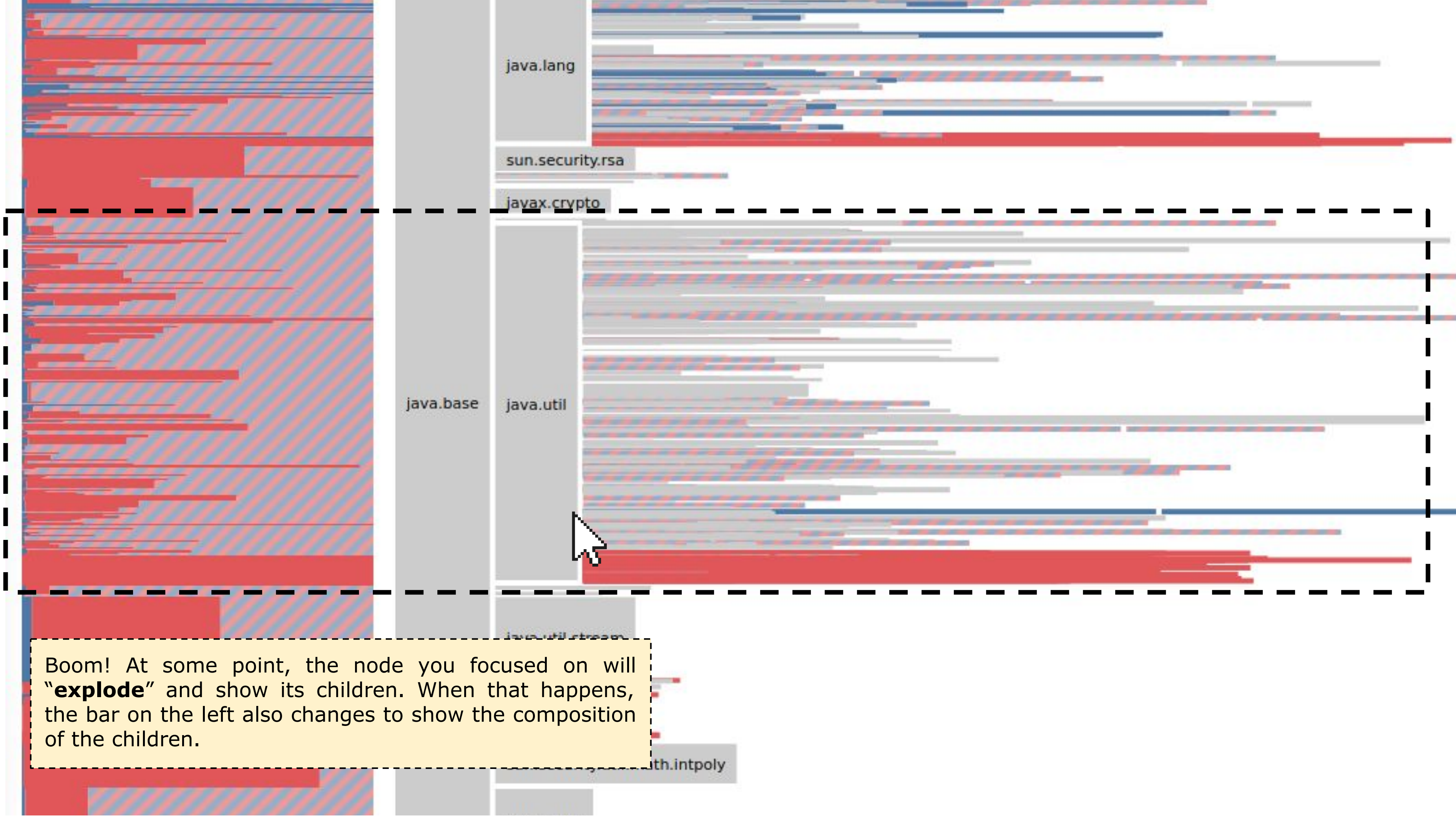
The red image uses a bigger chunk exclusively.

The vast majority of java.util is used by both the red and the blue image.

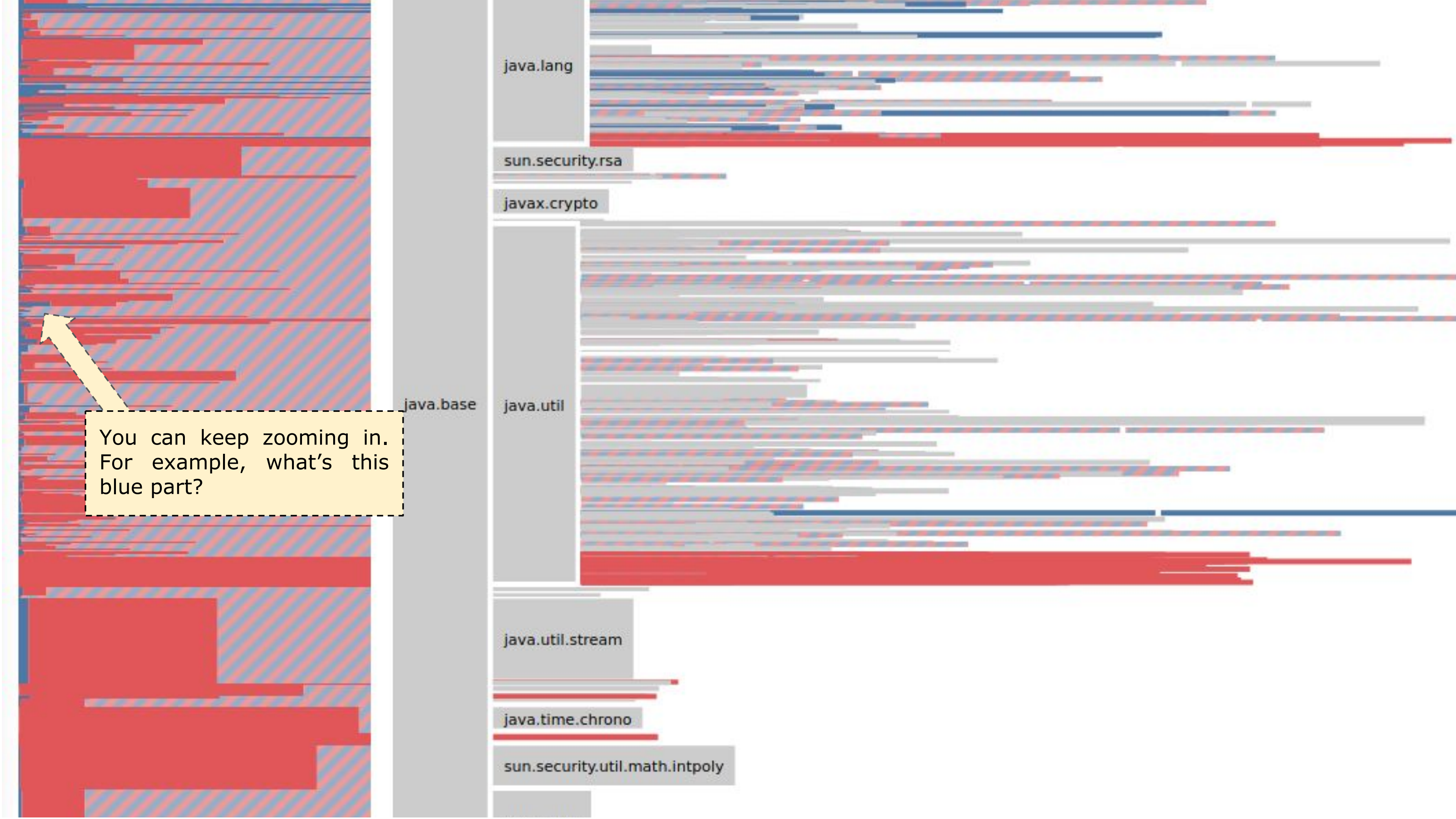


You are intrigued by something you see? The main mode of interaction with the Tree Line is **zooming**. Hover over the package you want to focus on and zoom in.

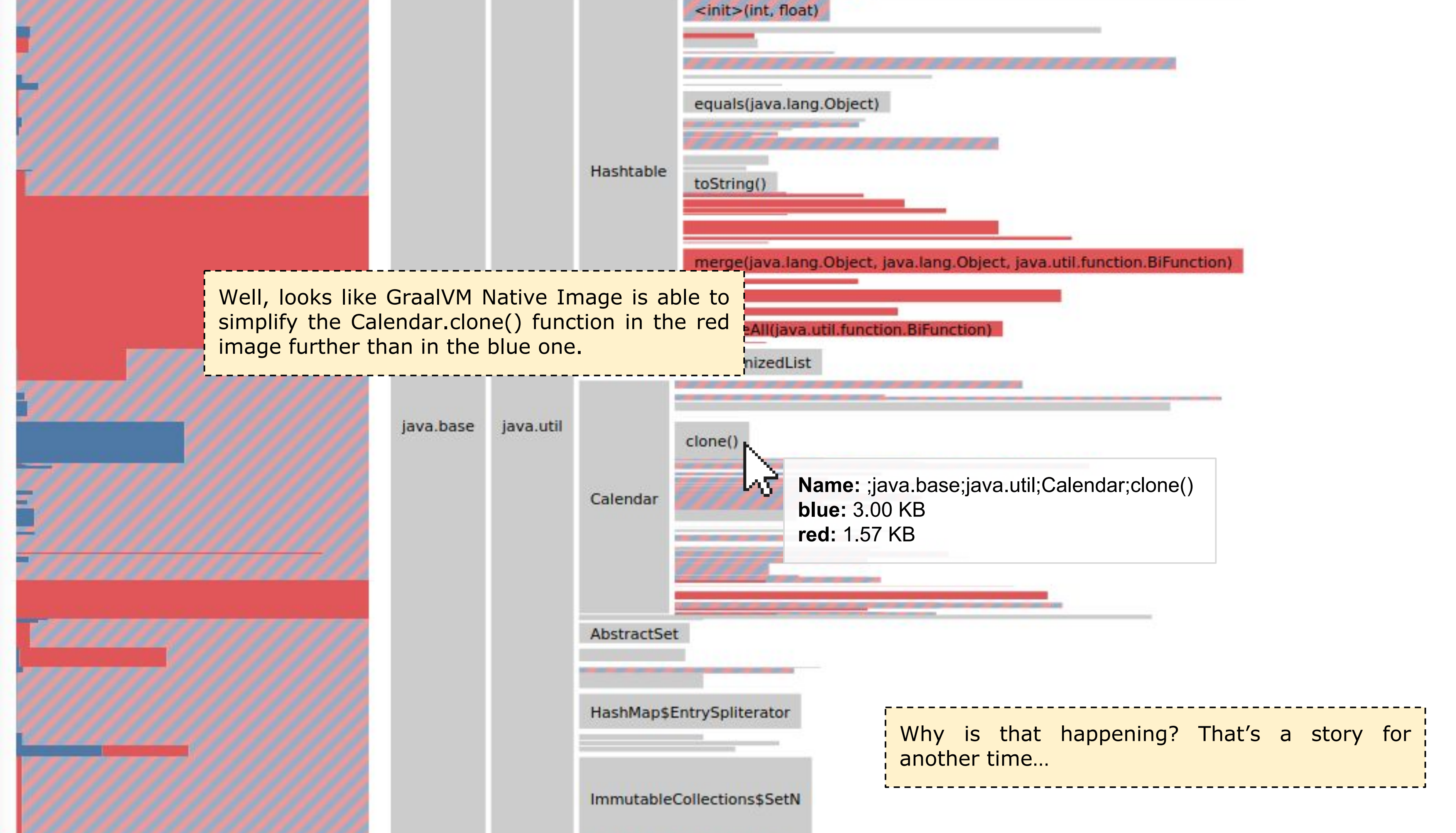
scroll to zoom in



Boom! At some point, the node you focused on will "**explode**" and show its children. When that happens, the bar on the left also changes to show the composition of the children.



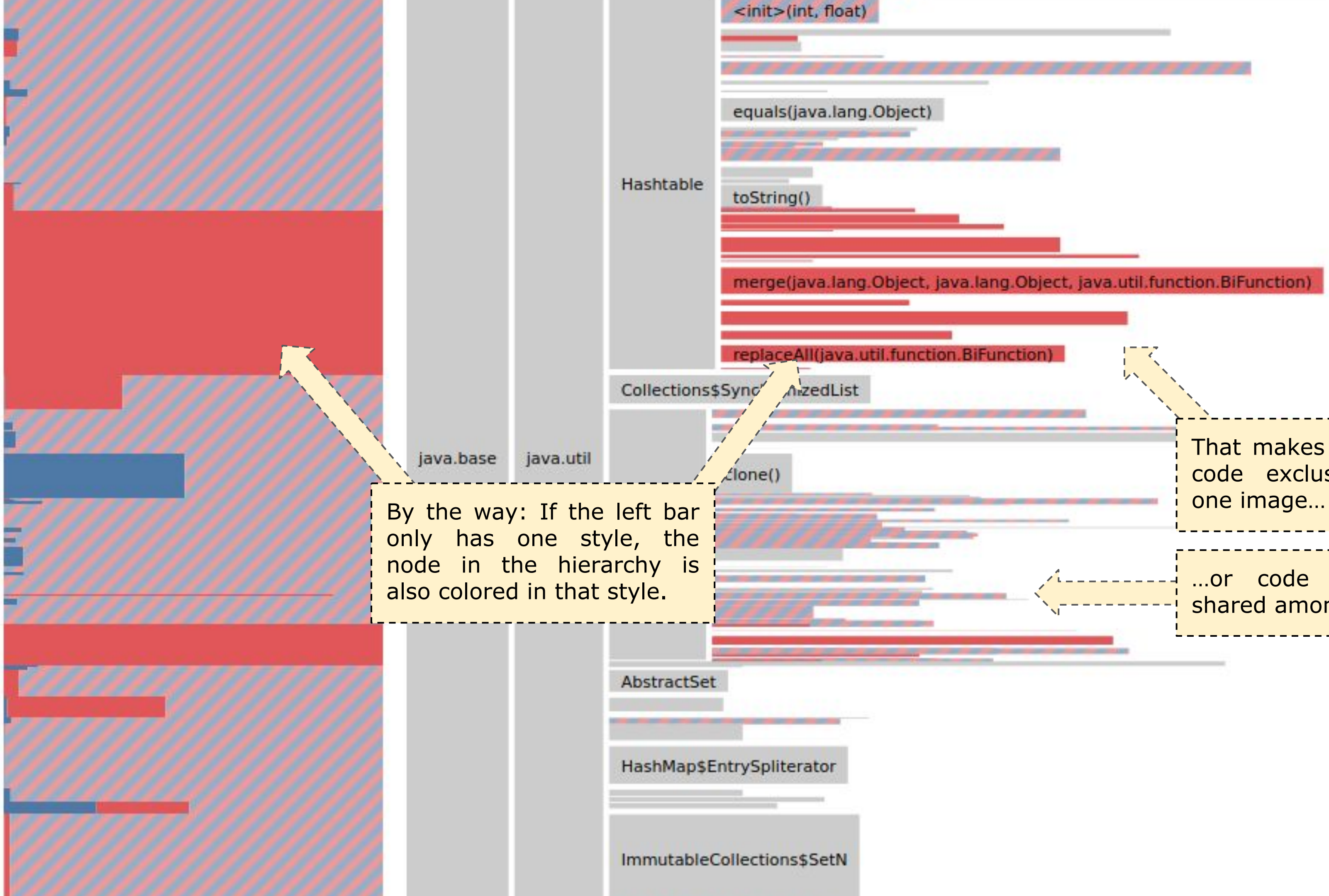
You can keep zooming in. For example, what's this blue part?



Name: ;java.base;java.util;Calendar;clone()
blue: 3.00 KB
red: 1.57 KB

Well, looks like GraalVM Native Image is able to simplify the Calendar.clone() function in the red image further than in the blue one.

Why is that happening? That’s a story for another time…

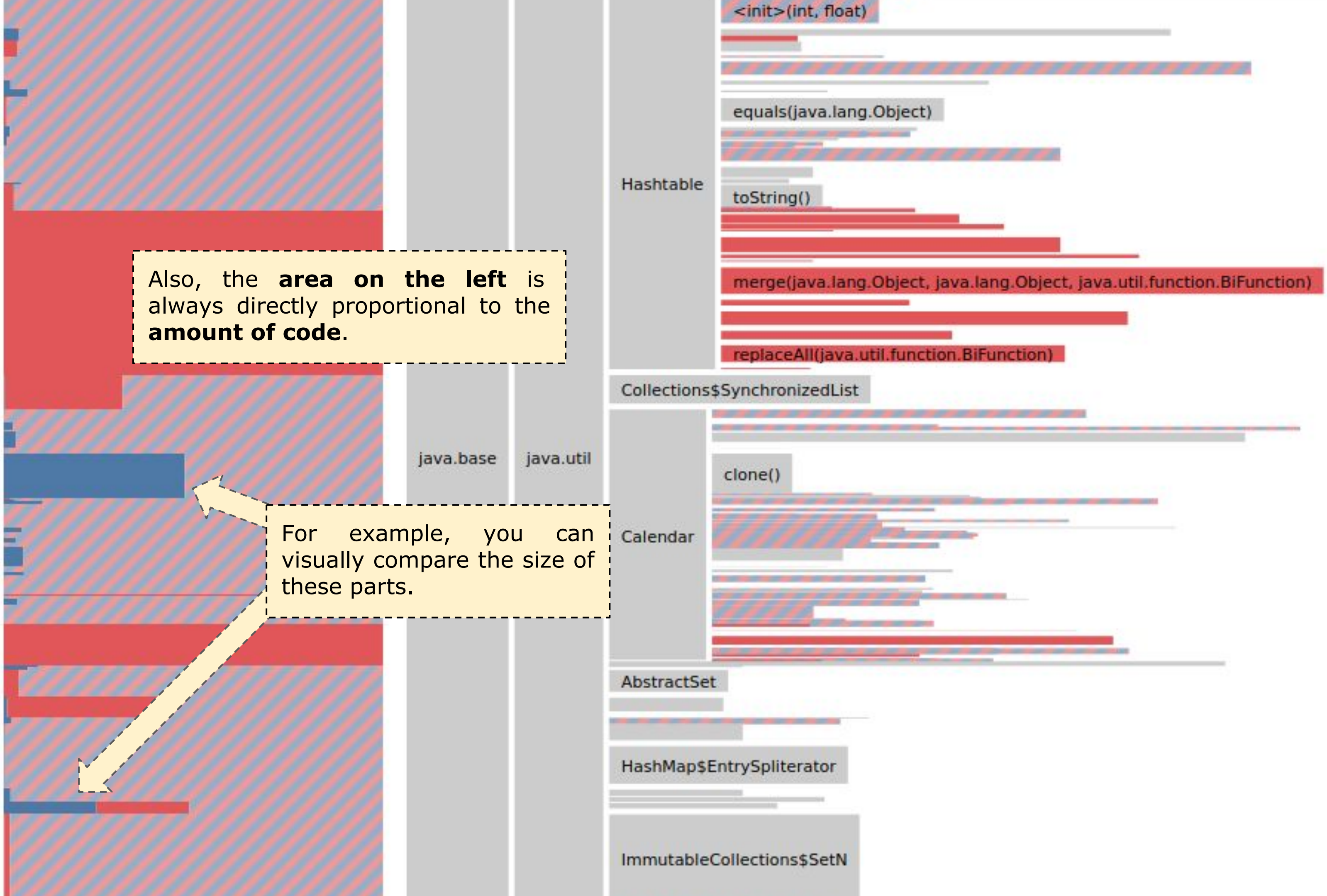


<init>(int, float)
equals(java.lang.Object)
Hashtable
toString()
merge(java.lang.Object, java.lang.Object, java.util.function.BiFunction)
replaceAll(java.util.function.BiFunction)
Collections$Sync nizedList
java.base
java.util
clone()
AbstractSet
HashMap$EntrySpliterator
ImmutableCollections$SetN

By the way: If the left bar only has one style, the node in the hierarchy is also colored in that style.

That makes it easy to spot code exclusively used in one image…

…or code that's entirely shared among all images.



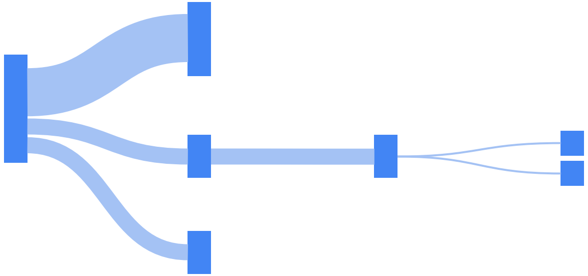
Also, the **area on the left** is always directly proportional to the **amount of code**.

For example, you can visually compare the size of these parts.

Sankey Tree

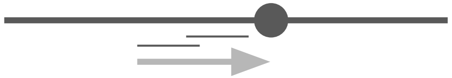
Layer Selection:

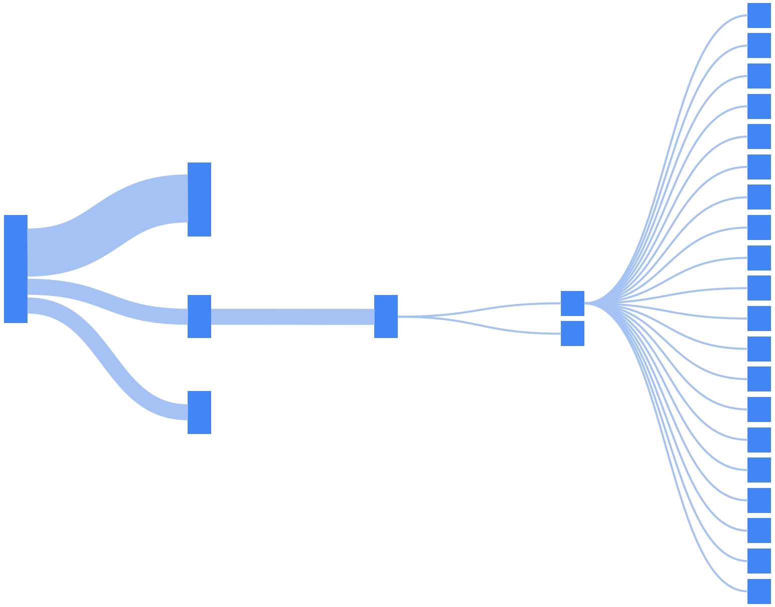
The default for the selected layer is “packages” due to performance reasons.



Layer Selection:

Changing the layer to a finer granularity allows for expanding the tree to a deeper level but sacrifices performance.





When rebuilding the sankey tree, a random branch is expanded to its deepest level to demonstrate the granularity change.

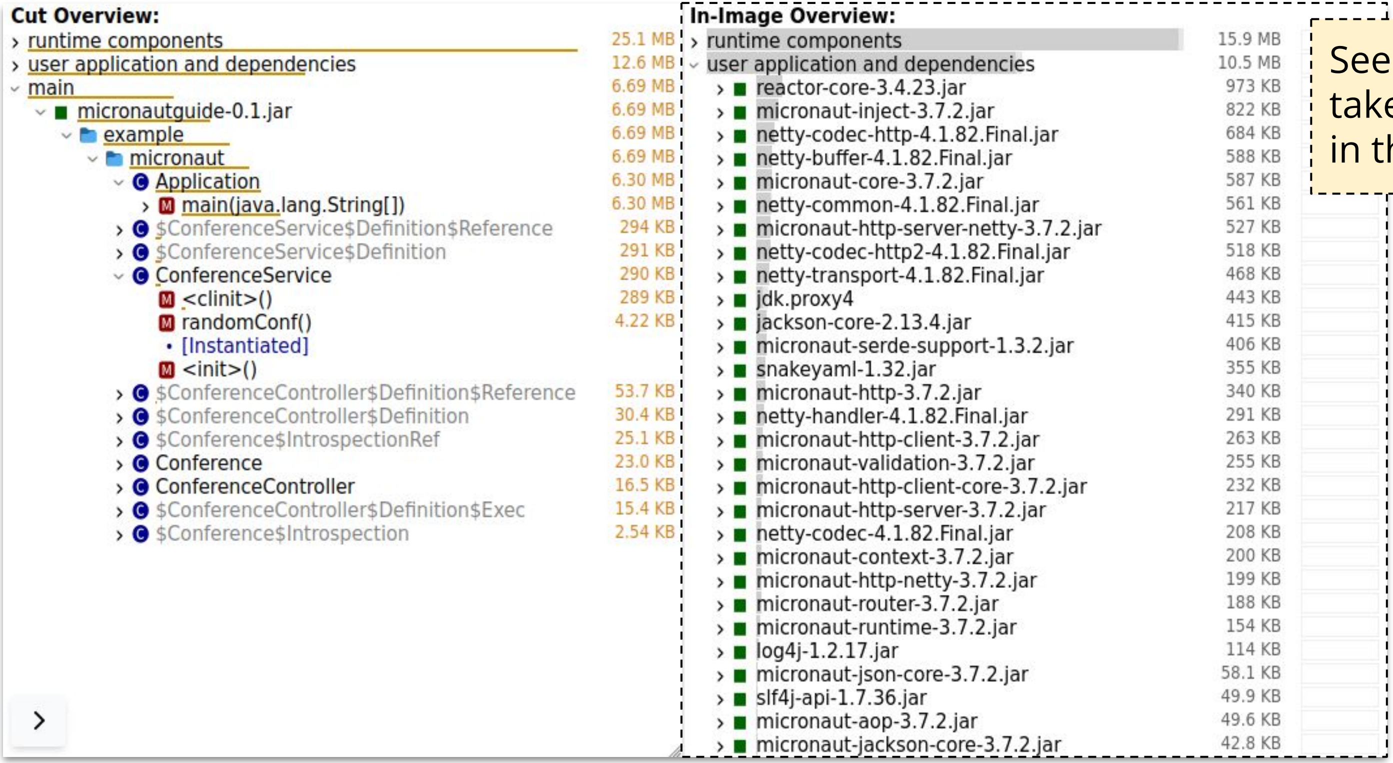
Filtering for a custom nodes selection:

Please, note that when filtering for a custom selection, branches where nodes other than the node **at the end** of a hierarchy level X is selected get filtered out and don’t appear on screen.

For example, only when node `ci` or `code` (=last node on the branch) from `;jdk.internal.vm.ci;jdk.vm.ci.code` is selected, the branch containing this node is filtered in. Selecting any other node here (e.g., `;jdk`), filters this branch out.

A node’s position in the hierarchy is indicated by its path, shown in the tooltip.

CutTool



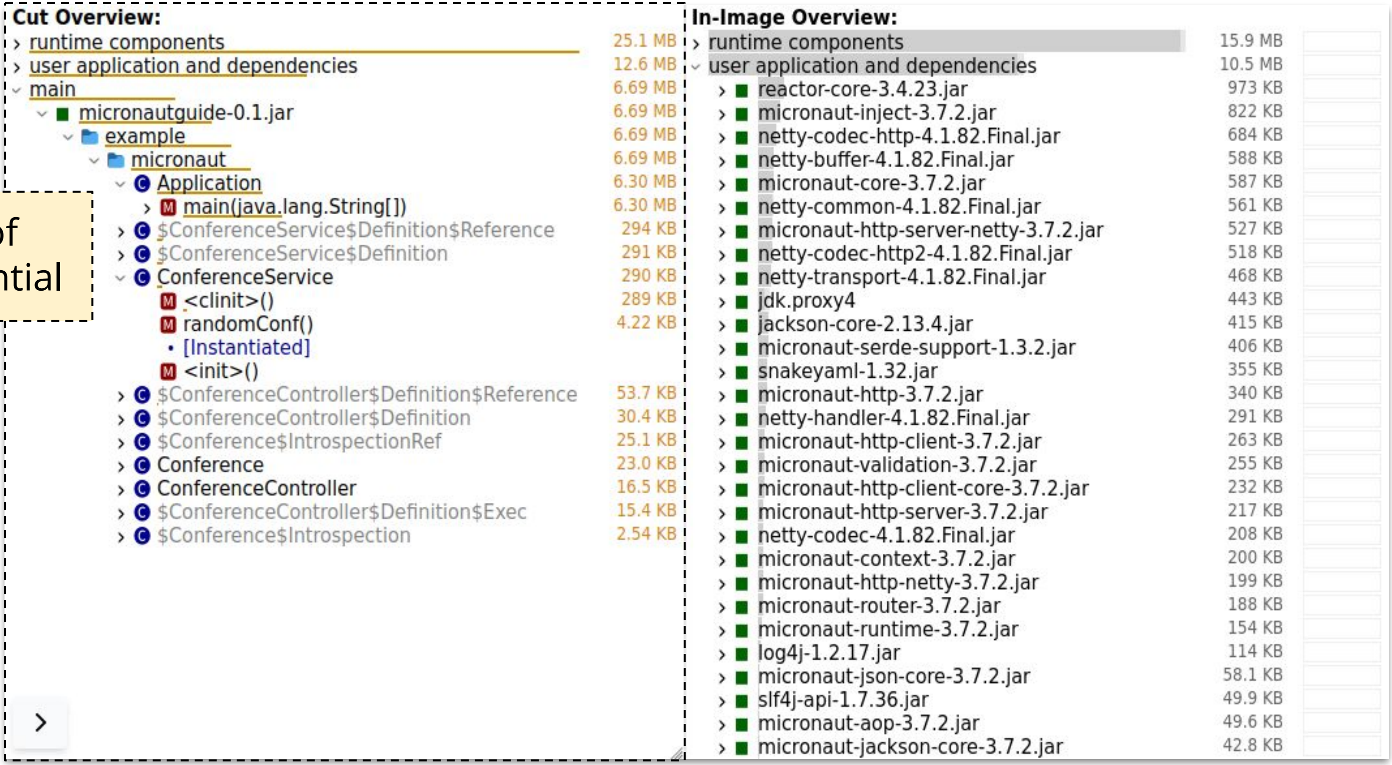
Cut Overview:

```
› runtime components                                    25.1 MB
› user application and dependencies                     12.6 MB
⌄ main                                                   6.69 MB
  ⌄ ■ micronautguide-0.1.jar                             6.69 MB
    ⌄ example                                            6.69 MB
      ⌄ micronaut                                        6.69 MB
        ⌄ C Application                                  6.30 MB
          › M main(java.lang.String[])                   6.30 MB
        › C $ConferenceService$Definition$Reference      294 KB
        › C $ConferenceService$Definition                291 KB
        ⌄ C ConferenceService                            290 KB
            M <clinit>()                                 289 KB
            M randomConf()                               4.22 KB
            • [Instantiated]
            M <init>()
        › C $ConferenceController$Definition$Reference   53.7 KB
        › C $ConferenceController$Definition             30.4 KB
        › C $Conference$IntrospectionRef                 25.1 KB
        › C Conference                                   23.0 KB
        › C ConferenceController                         16.5 KB
        › C $ConferenceController$Definition$Exec        15.4 KB
        › C $Conference$Introspection                    2.54 KB
```

In-Image Overview:

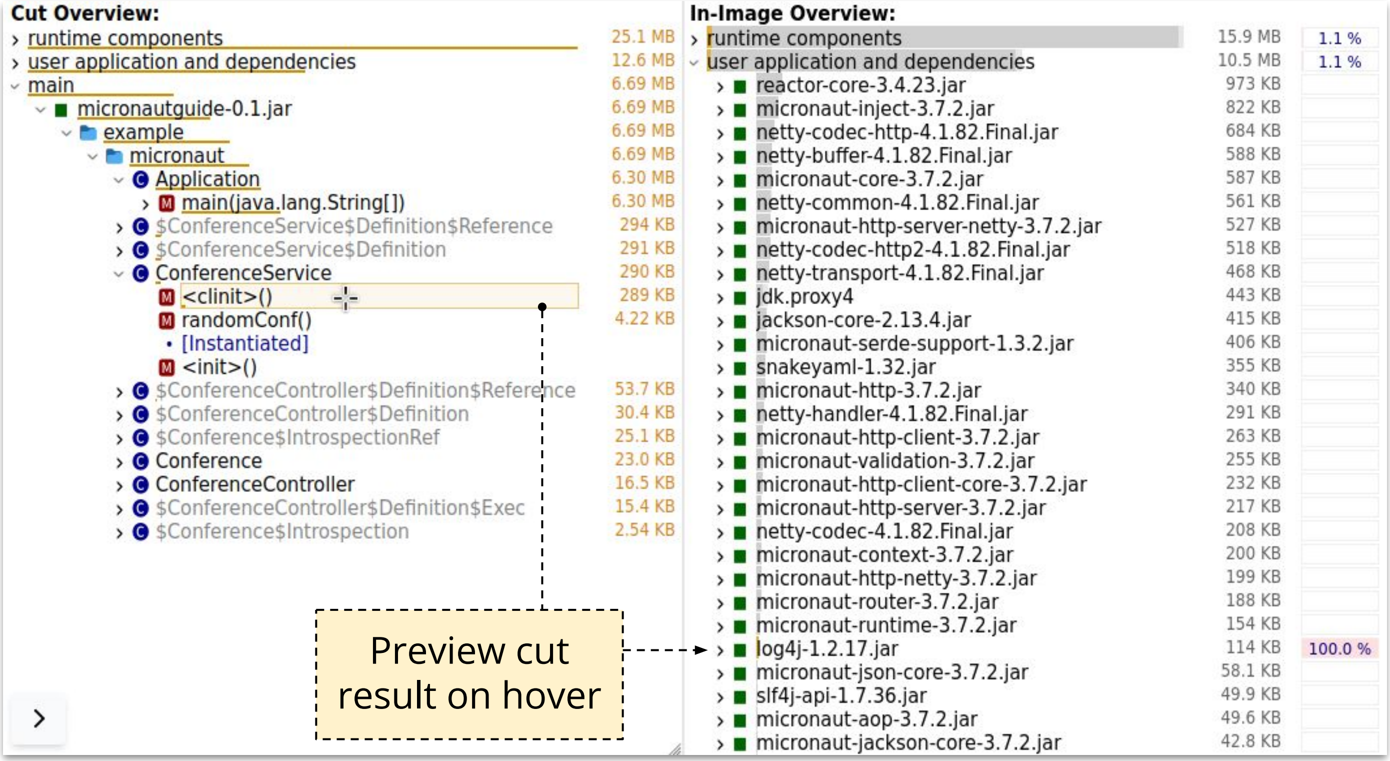
```
› runtime components                          15.9 MB
⌄ user application and dependencies           10.5 MB
  › ■ reactor-core-3.4.23.jar                 973 KB
  › ■ micronaut-inject-3.7.2.jar              822 KB
  › ■ netty-codec-http-4.1.82.Final.jar       684 KB
  › ■ netty-buffer-4.1.82.Final.jar           588 KB
  › ■ micronaut-core-3.7.2.jar                587 KB
  › ■ netty-common-4.1.82.Final.jar           561 KB
  › ■ micronaut-http-server-netty-3.7.2.jar   527 KB
  › ■ netty-codec-http2-4.1.82.Final.jar      518 KB
  › ■ netty-transport-4.1.82.Final.jar        468 KB
  › ■ jdk.proxy4                              443 KB
  › ■ jackson-core-2.13.4.jar                 415 KB
  › ■ micronaut-serde-support-1.3.2.jar       406 KB
  › ■ snakeyaml-1.32.jar                      355 KB
  › ■ micronaut-http-3.7.2.jar                340 KB
  › ■ netty-handler-4.1.82.Final.jar          291 KB
  › ■ micronaut-http-client-3.7.2.jar         263 KB
  › ■ micronaut-validation-3.7.2.jar          255 KB
  › ■ micronaut-http-client-core-3.7.2.jar    232 KB
  › ■ micronaut-http-server-3.7.2.jar         217 KB
  › ■ netty-codec-4.1.82.Final.jar            208 KB
  › ■ micronaut-context-3.7.2.jar             200 KB
  › ■ micronaut-http-netty-3.7.2.jar          199 KB
  › ■ micronaut-router-3.7.2.jar              188 KB
  › ■ micronaut-runtime-3.7.2.jar             154 KB
  › ■ log4j-1.2.17.jar                        114 KB
  › ■ micronaut-json-core-3.7.2.jar           58.1 KB
  › ■ slf4j-api-1.7.36.jar                    49.9 KB
  › ■ micronaut-aop-3.7.2.jar                 49.6 KB
  › ■ micronaut-jackson-core-3.7.2.jar        42.8 KB
```

See what takes space in the image



```
Cut Overview:                                                  In-Image Overview:
› runtime components                                 25.1 MB   › runtime components                        15.9 MB
› user application and dependencies                  12.6 MB   ⌄ user application and dependencies         10.5 MB
⌄ main                                               6.69 MB      › ■ reactor-core-3.4.23.jar               973 KB
   ⌄ ■ micronautguide-0.1.jar                        6.69 MB      › ■ micronaut-inject-3.7.2.jar            822 KB
      ⌄ example                                      6.69 MB      › ■ netty-codec-http-4.1.82.Final.jar     684 KB
         ⌄ micronaut                                 6.69 MB      › ■ netty-buffer-4.1.82.Final.jar         588 KB
            ⌄ C Application                          6.30 MB      › ■ micronaut-core-3.7.2.jar              587 KB
               › M main(java.lang.String[])          6.30 MB      › ■ netty-common-4.1.82.Final.jar         561 KB
            › C $ConferenceService$Definition$Reference  294 KB   › ■ micronaut-http-server-netty-3.7.2.jar 527 KB
            › C $ConferenceService$Definition         291 KB      › ■ netty-codec-http2-4.1.82.Final.jar    518 KB
            ⌄ C ConferenceService                     290 KB      › ■ netty-transport-4.1.82.Final.jar      468 KB
                 M <clinit>()                         289 KB      › ■ jdk.proxy4                            443 KB
                 M randomConf()                      4.22 KB      › ■ jackson-core-2.13.4.jar               415 KB
                 • [Instantiated]                                 › ■ micronaut-serde-support-1.3.2.jar     406 KB
                 M <init>()                                       › ■ snakeyaml-1.32.jar                    355 KB
            › C $ConferenceController$Definition$Reference 53.7 KB › ■ micronaut-http-3.7.2.jar            340 KB
            › C $ConferenceController$Definition     30.4 KB      › ■ netty-handler-4.1.82.Final.jar        291 KB
            › C $Conference$IntrospectionRef         25.1 KB      › ■ micronaut-http-client-3.7.2.jar       263 KB
            › C Conference                           23.0 KB      › ■ micronaut-validation-3.7.2.jar        255 KB
            › C ConferenceController                 16.5 KB      › ■ micronaut-http-client-core-3.7.2.jar  232 KB
            › C $ConferenceController$Definition$Exec 15.4 KB     › ■ micronaut-http-server-3.7.2.jar       217 KB
            › C $Conference$Introspection            2.54 KB      › ■ netty-codec-4.1.82.Final.jar          208 KB
                                                                  › ■ micronaut-context-3.7.2.jar           200 KB
                                                                  › ■ micronaut-http-netty-3.7.2.jar        199 KB
                                                                  › ■ micronaut-router-3.7.2.jar            188 KB
                                                                  › ■ micronaut-runtime-3.7.2.jar           154 KB
                                                                  › ■ log4j-1.2.17.jar                      114 KB
                                                                  › ■ micronaut-json-core-3.7.2.jar        58.1 KB
                                                                  › ■ slf4j-api-1.7.36.jar                 49.9 KB
                                                                  › ■ micronaut-aop-3.7.2.jar              49.6 KB
                                                                  › ■ micronaut-jackson-core-3.7.2.jar     42.8 KB
```

Prediction of cutoff potential



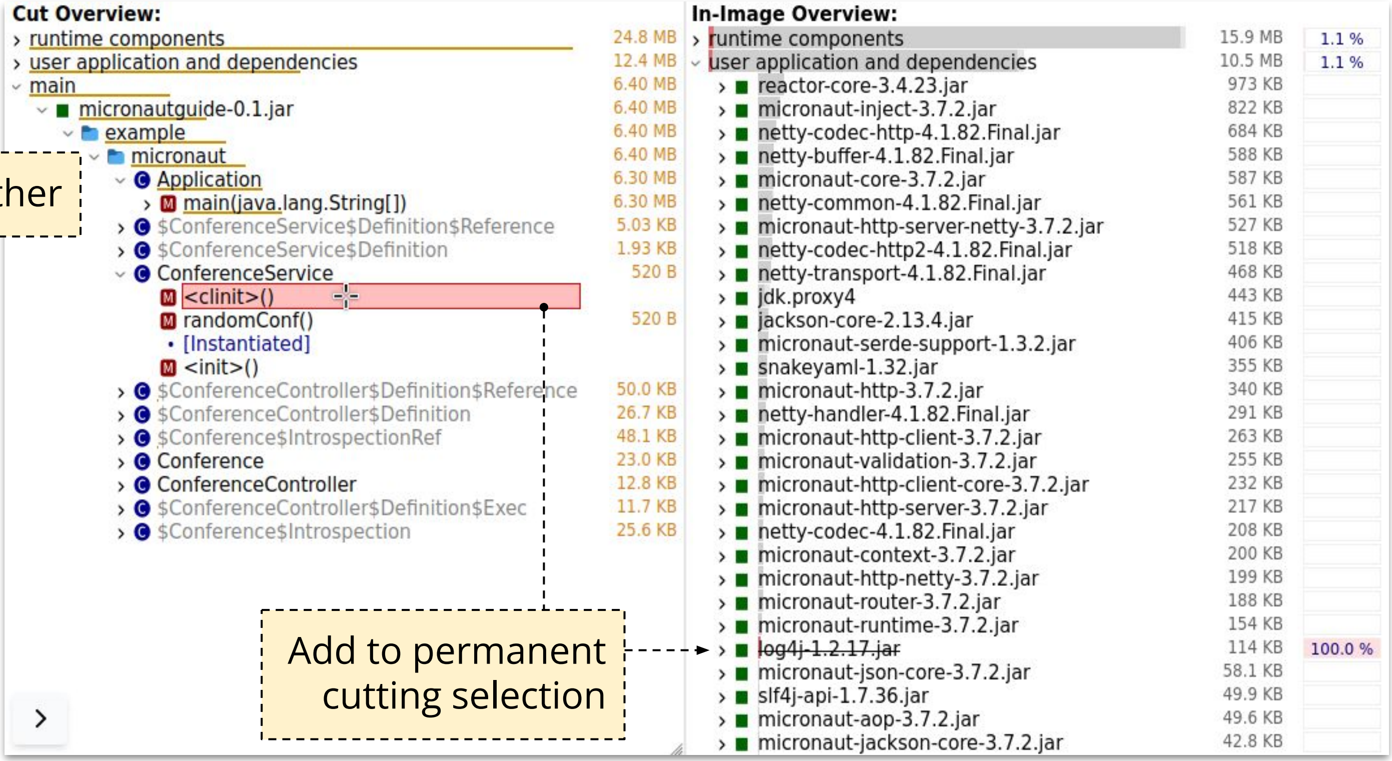
Cut Overview:

```
› runtime components                                          25.1 MB
› user application and dependencies                           12.6 MB
⌄ main                                                        6.69 MB
  ⌄ ■ micronautguide-0.1.jar                                  6.69 MB
    ⌄ example                                                 6.69 MB
      ⌄ micronaut                                             6.69 MB
        ⌄ C Application                                       6.30 MB
          › M main(java.lang.String[])                        6.30 MB
        › C $ConferenceService$Definition$Reference           294 KB
        › C $ConferenceService$Definition                     291 KB
        ⌄ C ConferenceService                                 290 KB
            M <clinit>()                                      289 KB
            M randomConf()                                    4.22 KB
            • [Instantiated]
            M <init>()
        › C $ConferenceController$Definition$Reference        53.7 KB
        › C $ConferenceController$Definition                  30.4 KB
        › C $Conference$IntrospectionRef                      25.1 KB
        › C Conference                                        23.0 KB
        › C ConferenceController                              16.5 KB
        › C $ConferenceController$Definition$Exec             15.4 KB
        › C $Conference$Introspection                         2.54 KB
```

In-Image Overview:

```
› runtime components                          15.9 MB    1.1 %
⌄ user application and dependencies           10.5 MB    1.1 %
  › ■ reactor-core-3.4.23.jar                 973 KB
  › ■ micronaut-inject-3.7.2.jar              822 KB
  › ■ netty-codec-http-4.1.82.Final.jar       684 KB
  › ■ netty-buffer-4.1.82.Final.jar           588 KB
  › ■ micronaut-core-3.7.2.jar                587 KB
  › ■ netty-common-4.1.82.Final.jar           561 KB
  › ■ micronaut-http-server-netty-3.7.2.jar   527 KB
  › ■ netty-codec-http2-4.1.82.Final.jar      518 KB
  › ■ netty-transport-4.1.82.Final.jar        468 KB
  › ■ jdk.proxy4                              443 KB
  › ■ jackson-core-2.13.4.jar                 415 KB
  › ■ micronaut-serde-support-1.3.2.jar       406 KB
  › ■ snakeyaml-1.32.jar                      355 KB
  › ■ micronaut-http-3.7.2.jar                340 KB
  › ■ netty-handler-4.1.82.Final.jar          291 KB
  › ■ micronaut-http-client-3.7.2.jar         263 KB
  › ■ micronaut-validation-3.7.2.jar          255 KB
  › ■ micronaut-http-client-core-3.7.2.jar    232 KB
  › ■ micronaut-http-server-3.7.2.jar         217 KB
  › ■ netty-codec-4.1.82.Final.jar            208 KB
  › ■ micronaut-context-3.7.2.jar             200 KB
  › ■ micronaut-http-netty-3.7.2.jar          199 KB
  › ■ micronaut-router-3.7.2.jar              188 KB
  › ■ micronaut-runtime-3.7.2.jar             154 KB
  › ■ log4j-1.2.17.jar                        114 KB     100.0 %
  › ■ micronaut-json-core-3.7.2.jar           58.1 KB
  › ■ slf4j-api-1.7.36.jar                    49.9 KB
  › ■ micronaut-aop-3.7.2.jar                 49.6 KB
  › ■ micronaut-jackson-core-3.7.2.jar        42.8 KB
```

Preview cut result on hover



Cut Overview:

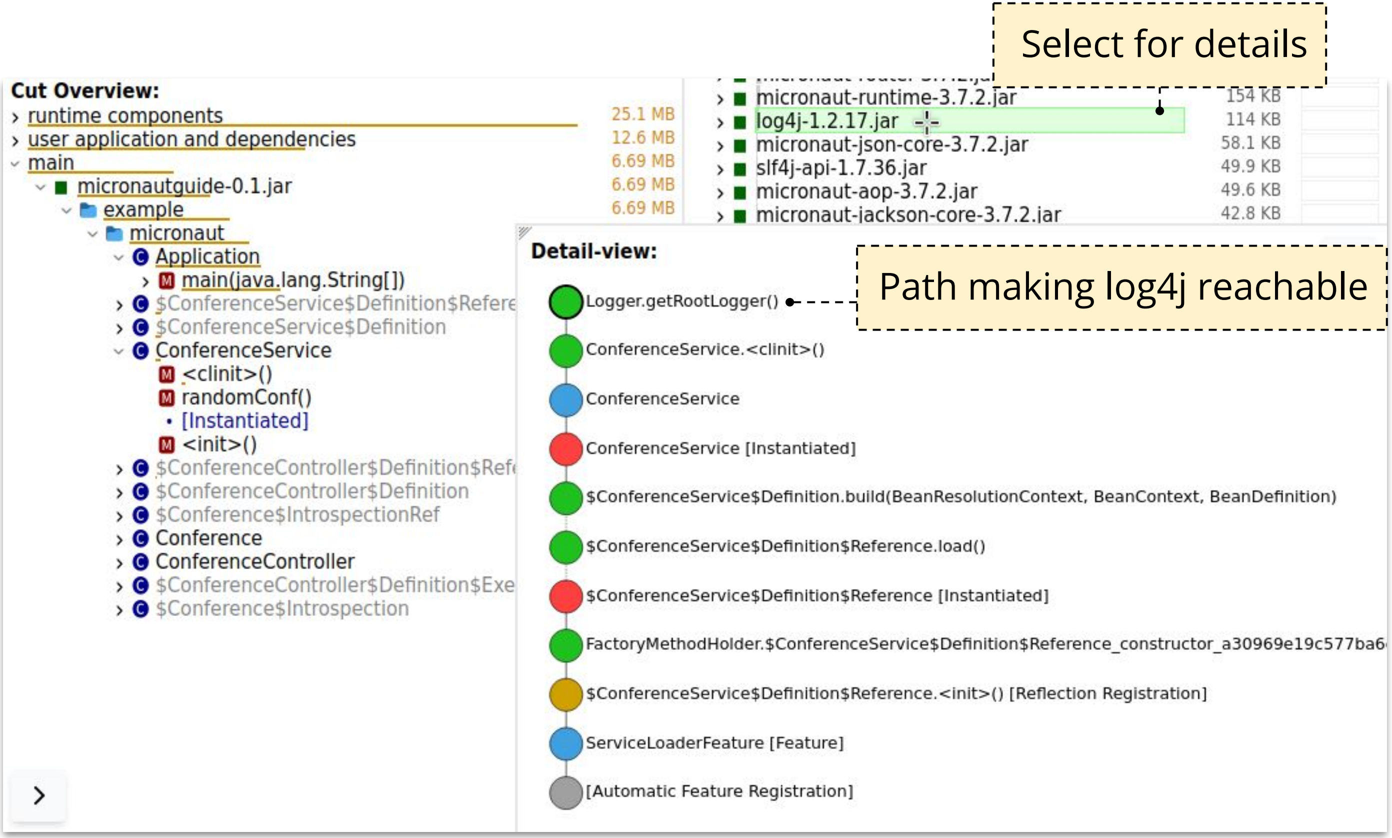
› runtime components	24.8 MB
› user application and dependencies	12.4 MB
⌄ main	6.40 MB
⌄ micronautguide-0.1.jar	6.40 MB
⌄ example	6.40 MB
⌄ micronaut	6.40 MB
⌄ Application	6.30 MB
› main(java.lang.String[])	6.30 MB
› $ConferenceService$Definition$Reference	5.03 KB
› $ConferenceService$Definition	1.93 KB
⌄ ConferenceService	520 B
<clinit>()	
randomConf()	520 B
• [Instantiated]	
<init>()	
› $ConferenceController$Definition$Reference	50.0 KB
› $ConferenceController$Definition	26.7 KB
› $Conference$IntrospectionRef	48.1 KB
› Conference	23.0 KB
› ConferenceController	12.8 KB
› $ConferenceController$Definition$Exec	11.7 KB
› $Conference$Introspection	25.6 KB

In-Image Overview:

› runtime components	15.9 MB	1.1 %
⌄ user application and dependencies	10.5 MB	1.1 %
› reactor-core-3.4.23.jar	973 KB	
› micronaut-inject-3.7.2.jar	822 KB	
› netty-codec-http-4.1.82.Final.jar	684 KB	
› netty-buffer-4.1.82.Final.jar	588 KB	
› micronaut-core-3.7.2.jar	587 KB	
› netty-common-4.1.82.Final.jar	561 KB	
› micronaut-http-server-netty-3.7.2.jar	527 KB	
› netty-codec-http2-4.1.82.Final.jar	518 KB	
› netty-transport-4.1.82.Final.jar	468 KB	
› jdk.proxy4	443 KB	
› jackson-core-2.13.4.jar	415 KB	
› micronaut-serde-support-1.3.2.jar	406 KB	
› snakeyaml-1.32.jar	355 KB	
› micronaut-http-3.7.2.jar	340 KB	
› netty-handler-4.1.82.Final.jar	291 KB	
› micronaut-http-client-3.7.2.jar	263 KB	
› micronaut-validation-3.7.2.jar	255 KB	
› micronaut-http-client-core-3.7.2.jar	232 KB	
› micronaut-http-server-3.7.2.jar	217 KB	
› netty-codec-4.1.82.Final.jar	208 KB	
› micronaut-context-3.7.2.jar	200 KB	
› micronaut-http-netty-3.7.2.jar	199 KB	
› micronaut-router-3.7.2.jar	188 KB	
› micronaut-runtime-3.7.2.jar	154 KB	
› ~~log4j-1.2.17.jar~~	114 KB	100.0 %
› micronaut-json-core-3.7.2.jar	58.1 KB	
› slf4j-api-1.7.36.jar	49.9 KB	
› micronaut-aop-3.7.2.jar	49.6 KB	
› micronaut-jackson-core-3.7.2.jar	42.8 KB	

Explore further

Add to permanent cutting selection



Select for details

Path making log4j reachable

```
Cut Overview:
› runtime components                              25.1 MB
› user application and dependencies               12.6 MB
⌄ main                                            6.69 MB
  ⌄ ■ micronautguide-0.1.jar                      6.69 MB
    ⌄ example                                     6.69 MB
      ⌄ micronaut
        ⌄ C Application
          › M main(java.lang.String[])
        › C $ConferenceService$Definition$Refere
        › C $ConferenceService$Definition
        ⌄ C ConferenceService
            M <clinit>()
            M randomConf()
            • [Instantiated]
            M <init>()
        › C $ConferenceController$Definition$Refe
        › C $ConferenceController$Definition
        › C $Conference$IntrospectionRef
        › C Conference
        › C ConferenceController
        › C $ConferenceController$Definition$Exe
        › C $Conference$Introspection
```

```
› ■ micronaut-runtime-3.7.2.jar          154 KB
› ■ log4j-1.2.17.jar                     114 KB
› ■ micronaut-json-core-3.7.2.jar        58.1 KB
› ■ slf4j-api-1.7.36.jar                 49.9 KB
› ■ micronaut-aop-3.7.2.jar              49.6 KB
› ■ micronaut-jackson-core-3.7.2.jar     42.8 KB
```

Detail-view:

- Logger.getRootLogger()
- ConferenceService.<clinit>()
- ConferenceService
- ConferenceService [Instantiated]
- $ConferenceService$Definition.build(BeanResolutionContext, BeanContext, BeanDefinition)
- $ConferenceService$Definition$Reference.load()
- $ConferenceService$Definition$Reference [Instantiated]
- FactoryMethodHolder.$ConferenceService$Definition$Reference_constructor_a30969e19c577ba6
- $ConferenceService$Definition$Reference.<init>() [Reflection Registration]
- ServiceLoaderFeature [Feature]
- [Automatic Feature Registration]