Quartz MuleSoft Support Guide

2024

Table of Contents

[1. Edge and LTS Releases for Mule 3](#_Toc172238626)

[1.1 Edge Channel 4](#_Toc172238627)

[1.2 LTS Channel 4](#_Toc172238628)

[1.3 Switch from an Edge Version to an LTS Version 4](#_Toc172238629)

[1.4 Mule Runtime Release Cadence Support 4](#_Toc172238630)

[1.5 Deployment Models and Application Lifecycle Management 5](#_Toc172238631)

[1.6 Mule Runtime Version Naming Changes 6](#_Toc172238632)

[1.7 Cloudhub, Cloudhub 2.0, and Runtime Fabric 7](#_Toc172238633)

[1.8 Hybrid Standalone 7](#_Toc172238634)

[2. Upgrading MuleSoft to Java 17: A Comprehensive Guide 9](#_Toc172238635)

[2.1 Why Upgrade to Java 17 with Mule 4.6? 9](#_Toc172238636)

[2.2 Plan Ahead with Our Release Calendar 9](#_Toc172238637)

[2.3 Navigating the Upgrade from Mule 4.4 to 4.6 10](#_Toc172238638)

[2.4 Mule Apps 11](#_Toc172238639)

[2.5 Key Changes to Note 11](#_Toc172238640)

[2.6 Testing Your Upgraded System 11](#_Toc172238641)

[2.7 Mule Runtime 4.6 and Upgrading to Java 17 11](#_Toc172238642)

[2.8 Additional Resources 12](#_Toc172238643)

[2.9 Java 17 Compatible Anypoint Connectors 13](#_Toc172238644)

[3 MuleSoft Java 8 to Java 17 upgrade 14](#_Toc172238645)

[3.1 Prerequisites 14](#_Toc172238646)

[3.2 Steps for Migration 14](#_Toc172238647)

[3.3 Additional Considerations 16](#_Toc172238648)

[3.4 Useful Resources 16](#_Toc172238649)

[4. Mule runtime version and release channel upgrade 24](#_Toc172238650)

[4.1 Pre-Upgrade Preparation 24](#_Toc172238651)

[4.2 Upgrade Process 24](#_Toc172238652)

[4.3 Post-Upgrade Steps 26](#_Toc172238653)

[5 Rollback Plan 27](#_Toc172238654)

[5.1 Prepare Rollback Plan 27](#_Toc172238655)

[5.2 Steps for CloudHub 2.0 27](#_Toc172238656)

[6 Mule Rollback Plan 29](#_Toc172238657)

[6.1 Prepare Rollback Plan 29](#_Toc172238658)

[6.2 Steps for CloudHub 2.0 29](#_Toc172238659)

# Edge and LTS Releases for Mule

Starting with Mule 4.5, MuleSoft introduces two new release channels, Edge and Long-term Support (LTS). Both release channels are available in all our deployment models: Anypoint Runtime Fabric, CloudHub, CloudHub 2.0, and Hybrid Standalone. Consider the following factors to determine the approach that best suits your requirements ahead of the Mule 4.4 End of Standard Support.

| **Channel Type** | **Release Cadence** | **Recommended Customer Profile** | **Differentiators** | **Support Coverage** |
| --- | --- | --- | --- | --- |
| Edge | Three times per year:   * February * June * October | * CloudHub and CloudHub 2.0 | * Includes new features on a frequent release cadence with shorter maintenance coverage. * Versions are not FedRAMP approved. | * 4 months of Standard Support * 4 months of Extended Support |
| LTS | Annual:   * February | On-premises:   * Hybrid Standalone * Runtime Fabric | * Includes new features introduced in prior Edge releases, along with new features introduced in the February Edge release. This release is maintained for an extended period over being the first to adopt new capabilities. * Versions are FedRAMP approved. | * 12 months of Standard Support * 12 months of Extended Support |

In February, both Edge and LTS release with the same underlying version, including a minor Mule runtime version and new features.

To review the new support policy that aligns with the new release periods and types of releases, see [MuleSoft Support Policy](https://www.mulesoft.com/legal/versioning-back-support-policy#mule-runtimes)

|  |
| --- |
| The monthly patching process remains unchanged. See [Update Patches to Mule](https://docs.mulesoft.com/release-notes/mule-runtime/patching-mule-versions). |

* 1. Edge Channel

Starting with Mule 4.5, MuleSoft releases Edge versions in February, June, and October. Use the Edge channel if you want to stay on top of the latest features and feel comfortable with a more frequent release cycle and auto-app upgrade cycle. The Edge channel includes the latest features and innovations available on the MuleSoft platform.

* 1. LTS Channel

Starting with Mule 4.6, MuleSoft releases Long-Term Support (LTS) versions annually in February. If you are on Mule 4.4, you are not required to upgrade to Mule 4.6. However, because Mule 4.5 releases in October 2023, Standard Support for Mule 4.4 ends in October 2024, followed by the End of Extended Support in October 2025. For CloudHub and CloudHub 2.0 customers, once Standard Support for Mule 4.4 ends, if you want to deploy new apps, you must do so on Mule 4.6 LTS or Mule 4.8 Edge. Use the LTS channel if you want to avoid maintaining new versions frequently. The LTS channel adopts new capabilities after they have been introduced via Edge releases.

* 1. Switch from an Edge Version to an LTS Version

You can switch from an Edge version to an LTS version and vice versa. However, when moving from Edge to LTS, you must choose a later LTS version. For example, if you choose Mule 4.6 LTS and you want to implement a specific new feature released on Mule 4.7 Edge, you can opt into that Edge release, then merge into Mule 4.9 LTS once it is available. MuleSoft does not support apps that are switched from Edge to an older LTS version. For example, switching an app from Mule 4.7 Edge to Mule 4.6 LTS might result in losing some of the new features in the older LTS version.

* 1. Mule Runtime Release Cadence Support

The following table shows the Mule runtime release cadence support:

| **Mule Version** | **Release Date** | **Java Version** | **End of Standard Support** | **End of Extended Support** |
| --- | --- | --- | --- | --- |
| **4.5 Edge** | **October 2023** | **8 and 11** | **February 2024** | **June 2024** |
| **4.6 LTS** | **February 2024** | **8, 11, and 17** | **February 2025** | **February 2026** |
| **4.6 Edge** | **February 2024** | **8, 11, and 17** | **June 2024** | **October 2024** |
| **4.7 Edge** | **June 2024** | **8, 11, and 17** | **October 2024** | **February 2025** |
| **4.8 Edge** | **October 2024** | **8, 11, and 17** | **February 2025** | **June 2025** |
| **4.9 LTS** | **February 2025** | **17** | **February 2026** | **February 2027** |
| **4.9 Edge** | **February 2025** | **17** | **June 2025** | **October 2025** |

|  |
| --- |
| If you are a CloudHub or a CloudHub 2.0 customer and you want to create or test Mule 4.6 applications in Anypoint Studio, the Studio version required is the next upcoming release in March. Meanwhile, MuleSoft enables you to deploy new apps in Mule runtime 4.5e version via API, Mule Maven plugin, and Anypoint Platform CLI starting February 14 until April 19. Auto-upgrade from Mule 4.5 Edge to Mule 4.6 Edge for CloudHub and CloudHub 2.0 applications will be performed in April instead of February. See [Temporary changes to auto-upgrade and new app deployment for Mule 4.5 Edge](https://help.mulesoft.com/s/article/Mule-Runtime-4-6-release-temporary-changes-to-auto-upgrade-and-new-app-deployment). |

* 1. Deployment Models and Application Lifecycle Management

Application lifecycle management depends on the deployment model.

| **Action** | **CloudHub and CloudHub 2.0** | **Runtime Fabric** | **Hybrid Standalone** |
| --- | --- | --- | --- |
| Deploy net new apps | Deploy to the latest minor-patch version available under Standard Support of each channel. | Deploy to any patch version of Mule runtime available under Standard or Extended Support. By default, the latest version of Mule runtime is selected. | Deploy while versions are in support period. |
| Auto-upgrade | * Monthly upgrade when a new patch version is available. * Edge channel apps are auto-upgraded within two weeks. * LTS channel apps are auto-upgraded after two months when a new minor version is available. | Applications aren’t auto-upgraded by MuleSoft without customer-initiated actions. | No auto-upgrade capabilities |
| Self-upgrade | The auto-upgrade occurs during a fixed period after MuleSoft releases the version and enforces the force-upgrade. During that period, you can do a manual upgrade.   * Edge: two weeks * LTS: two months | Always available | Using the [Mule Upgrade Tool](https://docs.mulesoft.com/mule-runtime/latest/mule-upgrade-tool) |
| Rollback | Available to the previously used version | Available to the previously used version | Available to the previous used version using the [Mule Upgrade Tool](https://docs.mulesoft.com/mule-runtime/latest/mule-upgrade-tool). |
| Restart apps | Always | Always | Always while the version is within the support period |
| Keep running applications | Until End of Extended Support | Always | Always |
| End of Extended Support | Shutdown running apps. Starting with Mule 4.7, CloudHub and CloudHub 2.0 applications in the Edge or LTS channels that are still running [End of Life versions](https://www.salesforce.com/content/dam/web/en_us/www/documents/legal/Agreements/versioning-back-support-policy.pdf) will undergo a forced upgrade to the latest version during the monthly upgrade window (See upgrade window for [CloudHub](https://docs.mulesoft.com/release-notes/cloudhub/cloudhub-runtimes-release-notes) and [CloudHub 2.0](https://docs.mulesoft.com/release-notes/cloudhub-2/cloudhub-2-runtimes-release-notes)). | Cannot deploy new apps | Can deploy new apps |
| Retirement | Mule apps do not reach End of Life because they are always auto-upgraded. | When Mule apps reach End of Life, you can still restart or redeploy them using the same runtime version the apps are currently running on. | When Mule apps reach End of Life, MuleSoft reserves the right to close the connection of the Mule runtime server to Anypoint Platform. |
| Supportability | All Mule apps are under Standard Support. | Supported while Mule apps are within the assigned support period. | Supported while Mule apps are within the assigned support period. |
| Redeployment / Applying changes / Upgrades | Supported until End of Extended Support. | Supported when a build tag is provided. If a build tag is not provided, the application defaults to the latest patch of the minor version being used. If a wrong, invalid, or unsupported build tag is provided, the deployment fails. | Supported under Hybrid Stansalone. |

* 1. Mule Runtime Version Naming Changes

The version naming convention depends on the deployment model you are using. A version increments:

* MAJOR when a release includes features that introduce breaking changes and backward incompatibility.
* MINOR when a release includes all new features keeping backward compatibility with previous minors.
* PATCH when a release includes bug fixes and security updates that include upgrades to libraries with reported vulnerabilities. We build a new runtime and do a full validation test.
* BUILD when a release includes changes related to Image/AMI, including OS changes, OS security updates, and changes in products outside the runtime. It doesn’t include any runtime changes.

## Cloudhub, Cloudhub 2.0, and Runtime Fabric

The Mule runtime versioning schema for the new release channels is:

*Major[numeric] . Minor[numeric] . Patch[numeric] : Build[numeric] Channel[e for edge, nothing for LTS]*

Each February, MuleSoft releases both an Edge and an LTS release with the same *Major.Minor* version. To distinguish the versions, they are represented as:

* Edge: 4.6.0:1e
* LTS: 4.6.0:1

Unlike the Hybrid Standalone customer profile, CloudHub, CloudHub 2.0, and Runtime Fabric require regular OS updates, hence the addition of the build enumeration in the full runtime version schema.

Here are examples of the version numbers:

* Edge: 4.5.0:1e
* Edge: 4.6.0:1e
* LTS: 4.6.0:1

## Hybrid Standalone

The Mule runtime versioning changes from date-based, for example, 4.4.0-20230317 to semVer as:

*Major.Minor.Patch*

Here are examples of the version numbers:

* 4.5.1
* 4.5.2
* 4.5.3

Mule runtime does not introduce a semVer increment if there is a month with no fixes.

Here is an example of the versioning schema using different patch and builds:

|  |  |  |  | **Hybrid Standalone** | **CloudHub - AMI / CloudHub 2.0 / Runtime Fabric - Docker image** |
| --- | --- | --- | --- | --- | --- |
| **Case** | **Release Date** | **Description** | **Patch/Build** | **Mule Runtime** | **Runtime Manager/Maven/API Tag for New Deployment** |
| 1 | Oct 3, 2023 | First release of 4.5.0 | 0th patch and 1st build | 4.5.0 | 4.5.0:1e |
| 2 | Nov 7, 2023 | Second release of 4.5 | 1st patch and 1st build | 4.5.1 | 4.5.1:1e |
| 3 | Nov 7, 2023 | Another build on same day for CloudHub, CloudHub 2.0, and Runtime Fabric | 1st patch and 2nd build | N/A | 4.5.1:2e |
| 4 | Nov 7, 2023 | OS updates only for CloudHub, CloudHub 2.0, and Runtime Fabric | 1st patch and 3rd build | N/A | 4.5.1:3e |
| 5 | Dec 5, 2023 | Second patch version of 4.5 | 2nd patch and 1st build | 4.5.2 | 4.5.2:1e |
| 6 | Jan 2, 2024 | Hot fix for a Mule runtime regression | 3rd patch & 1st build | 4.5.3 | 4.5.3:1e |
| 7 | Feb 6, 2024 | Release of new minor | 0th patch and 1st build | 4.6.0 | 4.6.0:1e (Edge) / 4.6.0:1 (LTS) |

The Mule runtime versioning schema uses the following conventions:

* Patch number in schema
  + In the schema 4.5.X:2e, the patch number is the X.
  + The patch number starts from 0, introducing a new minor version.
  + The patch number increments when the release introduces new code changes, including hotfixes to regressions or other bug fixes.
* Build number in schema.
  + In the schema 4.5.1:Ye, the build number is Y.
  + The build number starts from 1, introducing the first AMI or Docker image build of the patch.
  + The build number increments whenever the release introduces a new build of the AMI or Docker image for the same Mule runtime version. This increment does not indicate code changes to Mule runtime.

# Upgrading MuleSoft to Java 17: A Comprehensive Guide

MuleSoft is adopting Java’s long-term-support (LTS) release cadence, beginning with Mule runtime 4.6, which adds support for Java 17 LTS.

This change has no immediate impact to Mule applications that are running on currently supported Mule versions, or on applications that are upgraded to Mule runtime 4.6. Existing Java 8 apps will continue to run on Java 8 until February 2025.

Utilize this article and linked resources to ensure a smooth transition to Java 17 before February 2025.

## Why Upgrade to Java 17 with Mule 4.6?

Boosted Performance:

* Java 17 introduces performance improvements such as quicker startup times, reduced garbage collection delays, and better handling of large applications.
* Experience a noticeable enhancement in your application's performance.

Strengthened Security:

* Java 17 brings critical security updates and fixes.
* Upgrading means your applications are protected with the latest defenses against vulnerabilities.

Assured Long-Term Support:

* Java 17 is an LTS release, offering extended support for a stable and reliable foundation, especially for enterprise applications with long lifecycles.

## Plan Ahead with Our Release Calendar

MuleSoft provides a release calendar to help you schedule your upgrade without immediate impact on your Mule applications. This gives you ample time to prepare for a smooth transition.

A screenshot of a calendar

Description automatically generated

Begin your transition with the Mule Runtime 4.6 release, which supports both Java 8 and Java 17 LTS. Anypoint Studio 7.17 is ready to assist with this upgrade, offering embedded support for both Java versions. Download Anypoint Studio 7.17.

MuleSoft has also updated all connectors for Java 17 compatibility. Check out the Java 17-compatible Anypoint Connectors knowledge base article for more information.

## Navigating the Upgrade from Mule 4.4 to 4.6

Strategize Your Upgrade:

* Conduct an analysis of your current Mule 4.4 setup to pinpoint compatibility issues or deprecated elements.
* Create a comprehensive upgrade plan, including testing, rollback strategies, and post-upgrade checks.
* Schedule downtime and inform stakeholders to minimize business disruption.
* Implementing the Upgrade:
* Review MuleSoft's official documentation for upgrade guidance.
* Test your integration flows, connectors, and APIs in a controlled environment before going live.
* Work closely with your teams to manage the upgrade process effectively.
* Check upgrade information for each product:
  + [**Mule Runtime**](https://docs.mulesoft.com/general/java-support#mule-runtime)
  + [**Anypoint Connectors**](https://docs.mulesoft.com/general/java-support#anypoint-connectors-and-modules) & [**Java 17-Compatible Anypoint Connectors**](https://help.salesforce.com/s/articleView?id=000782248&type=1&language=en_US)
  + [**Anypoint Custom Connectors**](https://docs.mulesoft.com/general/java-support#custom-connectors)
  + [**Anypoint Studio**](https://docs.mulesoft.com/general/java-support#anypoint-studio)
  + [**Policies and API Proxies**](https://docs.mulesoft.com/general/upgrade-policies-proxies)
  + [**Mule Maven Plugin**](https://docs.mulesoft.com/general/java-support#mule-maven-plugin)
  + [**Dataweave**](https://docs.mulesoft.com/general/java-support#dataweave)

## Mule Apps

You can update, test, and redeploy most of your currently running Mule apps to use Java 17. However, any apps and connectors that are running custom Java code require additional work to certify those components. Before you upgrade your integration apps or Mule Gateway policies and proxies to Java 17, you must update all extensions, modules, and connectors used within those apps and policies to Java 17.

To ensure your API proxies or Mule apps are protected when upgrading, upgrade your API policies before upgrading your API proxies or Mule apps.

## Key Changes to Note

* Major updates to accommodate Java 17 and GUI changes require modifications to project POM files.
* Update your POM file with the new Java version and release channel:

<javaVersion>17</javaVersion>

<releaseChannel>LTS</releaseChannel>

* Recompile all connectors and update Maven plugin and Munit versions.
* Reconfigure API Manager policies post-deployment.

## Testing Your Upgraded System

* Thoroughly test all components to ensure they operate correctly after the upgrade.
* Allow ample time for QA testing and involve stakeholders to confirm performance and functionality standards are met.

## Mule Runtime 4.6 and Upgrading to Java 17

[**Watch this video**](https://videos.mulesoft.com/watch/nadaMTPizVjMyZwpmJJmJo) for an overview of the runtime update and advice on upgrading to Java 17.

## Additional Resources

Refer to the following for more information about MuleSoft Java 17upgrade:

* [**MuleSoft Java 17 Upgrade FAQ**](https://help.salesforce.com/s/articleView?id=000396936&type=1&language=en_US) - Consult this article for answers to frequently asked questions about upgrading to Java 17.
* [**Java Support**](https://docs.mulesoft.com/general/java-support) - Review this documentation for detailed guidance on the upgrade process and performance considerations.
* [**Anypoint Studio 7.17 – Easier Support for Java 17**](https://blogs.mulesoft.com/news/anypoint-studio-7-17/) - Read this blog to learn how Anypoint Studio 7.17 enhances support for Java 17, improving IDE stability and performance.
* [**Upgrading Java for Policies and API Proxies**](https://docs.mulesoft.com/general/upgrade-policies-proxies) - Refer to this documentation for steps to upgrade your policies and API proxies to be compatible with Java 17.
* [**Java 17 specific Known Issues**](https://issues.salesforce.com/#q=(java%2017%20OR%20JDK17%20)%20AND%20MuleSoft&sortCriteria=%40sfcreateddate%20descending&numberOfResults=50) - Refer to this site for any Java 17 MuleSoft known issues.
* [**#MuleSoftJava17Upgrade**](https://trailhead.salesforce.com/trailblazer-community/topics/mulesoftjava17upgrade?sort=LAST_MODIFIED_DATE_DESC) -  Engage with the Trailblazer Community to ask questions and share insights about the Java 17 upgrade.
* [**Java 17 Compatible Anypoint Connectors**](https://help.salesforce.com/s/articleView?id=000782248&type=1&language=en_US) - Access this article for a comprehensive list of MuleSoft Anypoint connectors that are Java 17 compatible.
* [**Upgrade Your Mule: Best Practices for Upgrading to Mule 4.6**](https://blogs.mulesoft.com/news/upgrading-to-mule-4-6/) - Follow this blog for best practices on upgrading to Mule 4.6, which fully supports Java 17 LTS.
* [**Upgrading Java for Custom Connectors (Customers)**](https://docs.mulesoft.com/general/customer-connector-upgrade) - Refer to this documentation to upgrade, test, and release your custom connectors for Java 17 to ensure compatibility within the MuleSoft ecosystem.

 Keeping your Mule versions up-to-date is vital for leveraging the full potential of your integration platform. Regular upgrades grant you access to the latest features, enhancements, and fixes, bolstering your system's performance and security. Stay compatible, reduce risks, and maintain a robust integration environment with MuleSoft.

## Java 17 Compatible Anypoint Connectors

The following are the list of MuleSoft Anypoint connectors which are Java 17 compatible.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# MuleSoft Java 8 to Java 17 upgrade

Migrating from Java 8 to Java 17 in Mule **4.6+** involves several steps and considerations to ensure compatibility and leverage the new features and improvements provided by Java 17. Here’s a comprehensive guide to help us through the migration process:

## Prerequisites

* **Mule Runtime Compatibility**: Ensure that the Mule **4.6+** and above runtime version we are using supports Java 17. MuleSoft documentation and release notes will have this information. Apps and policies that upgrade to Mule Runtime 4.9 (Feb 2025) will only support Java 17 and **will not support Java 8,** enables customers to leverage Java 17 capabilities.

1. **Backup**: Always backup/ export your existing Mule applications and configurations before making any changes.

## Steps for Migration

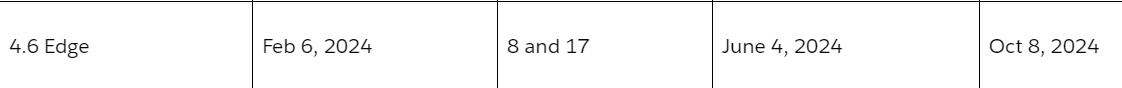
**1. Update the Java Version**

* Install Java 17 on your system / Or Use Anypoint Studio Embedded Java 17 version.
* Update your environment variables (JAVA\_HOME and PATH) to point to the Java 17 installation. (For Embedded JAVA Version we don’t have to set java environment variables).

**2. Update Mule Runtime**

* If required, update your Mule runtime to a version that supports Java 17. This can be done through Anypoint Studio or your deployment environment (such as MuleSoft CloudHub or on-premises servers).





**3. Update Project Configuration**

* In your Mule projects, update the **pom.xml** (for Maven-based projects) to specify Java 17 as the target version.
* Example for **pom.xml**:

xml

Copy code

<properties>

<mule.maven.plugin.version>3.3.5</mule.maven.plugin.version>

<mule.runtime.version>4.6.1</mule.runtime.version>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

(**Note**: Since In our case we don’t have CI/CD pipeline file, so we can only change the java version in cloudhub inside runtime manager)

**4. Update Custom Code**

* If your Mule application includes custom Java code, review and update it to ensure compatibility with Java 17.
* Look for deprecated APIs and features that have been removed in Java 17.
* Refactor code to use new Java 17 features where applicable, such as records, pattern matching, and sealed classes.

**Note**: For our all the applications we don’t have any custom Java code

**5. Update Dependencies**

* Update your project dependencies to versions that are compatible with Java 17.
* Pay special attention to third-party libraries and ensure they are updated to versions that support Java 17.

Note: Since we are already using latest runtime version so that all the projects’ libraries are compatible with java 17.

**6. Testing**

* Thoroughly test your Mule applications to ensure they work as expected with Java 17.
* Perform unit testing, integration testing, and user acceptance testing (UAT) to validate the functionality.

**7. Performance Tuning**

* Monitor the performance of your applications after migration.
* Java 17 may have different performance characteristics compared to Java 8, so adjust JVM parameters and application configurations as needed.

Note: We don’t have JVM for our existing project to apply the migration.

**8. Deployment**

* Deploy your updated Mule applications to your target environments.
* Monitor the applications for any issues post-deployment.

## Additional Considerations

* **Deprecation and Removal**: Java 17 may have deprecated or removed some features that were present in Java 8. Review the migration guides provided by Oracle to identify these changes.
* **Security Updates**: Java 17 includes security improvements and updates. Ensure that your applications adhere to the latest security practices.
* **Documentation and Resources**: Refer to MuleSoft documentation, Oracle’s Java documentation, and other migration guides for additional insights and best practices.

## Useful Resources

* MuleSoft Documentation
* [Java 17 Release Notes](https://www.oracle.com/java/technologies/javase/17-relnote-issues.html)
* [Java Migration Guides](https://www.oracle.com/java/technologies/migration-guide.html)
* <https://help.salesforce.com/s/articleView?id=000782248&type=1>

By following these steps, we can effectively migrate our Mule 4 applications from Java 8 to Java 17 and take advantage of the new features and improvements provided by the latest Java version.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A white sheet with black lines

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A screenshot of a graph

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a white sheet

Description automatically generated

A screenshot of a white sheet

Description automatically generated

A screenshot of a white sheet

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A white rectangular object with black lines

Description automatically generated with medium confidence

# Mule runtime version and release channel upgrade

Upgrading the runtime version in MuleSoft from **Edge 4.6.4 to LTS 4.6.5** involves several steps to ensure a smooth transition for our existing APIs. Here's a step-by-step guide:

## Pre-Upgrade Preparation

1. **Backup Existing Configurations:**
   * Ensure we have a backup of all our Mule applications, configurations, and any customizations. Export the jar file as a backup and use GitHub repo for version control and the updated code check-in.
2. **Review Release Notes:**
   * Read the release notes for MuleSoft 4.6.5 LTS to understand new features, bug fixes, and potential breaking changes. We have a documentation on LTS features.
3. **Check Compatibility:**
   * Verify that your applications and any third-party connectors or plugins are compatible with Mule LTS 4.6.5. Check dependencies and module in POM file if they need to upgrade.
4. **Environment Setup:**
   * Ensure we have a development or testing environment to test the upgrade before applying it to production.

## Upgrade Process

1. **Download/update New Runtime:**
   * Download/update Mule Runtime 4.6 from the MuleSoft exchange or Anypoint Platform.
2. **Update Mule platform Runtime:**
   * **CloudHub 2.0:** Update the runtime version in the CloudHub 2.0 deployment settings.
     + Log in to Anypoint Platform.
     + Navigate to Runtime Manager.
     + Select the application you want to update.
     + In the application settings, change the runtime version and release channel from Edge 4.6.4 to LTS 4.6.5.

**1.Anypoint platform runtime manager (current version and release channel):**

A screenshot of a computer

Description automatically generated

2.**Anypoint platform runtime manager (change to new version and release channel):**

A screenshot of a computer

Description automatically generated

3.**Anypoint platform runtime manager in the application file, upload the new jar file if the dependency and module upgraded.**

A screenshot of a computer

Description automatically generated

4.**Checkin the old code or exported jar file into github repository.**

1. **Update Applications:**
   * Open each Mule application in Anypoint Studio.
   * Update the **mule-deploy.properties** file to reflect the new runtime version if we have it. Or update the dependency and module in the POM file if the upgrade is required.
   * Test each application locally with the new runtime version and dependencies.
2. **Test:**
   * Deploy the applications to a dev/tst environment.
   * Perform testing after CloudHub 2.0 runtime deployment to ensure all functionalities are working as expected.
   * Pay special attention to custom connectors, DataWeave scripts, and any external integrations if we have it. Unfortunately, we don’t have custom connectors to consider.

## Post-Upgrade Steps

1. **Deploy to Production:**
   * Once testing is successful, deploy the applications to the production environment.
   * Monitor the applications closely for any issues.
2. **Update Documentation:**
   * Update any internal documentation to reflect the changes made during the upgrade.
3. **Notify Stakeholders:**

Inform all relevant stakeholders about the upgrade and any potential changes in behavior

# Rollback Plan

## Prepare Rollback Plan

* + Have a rollback plan in place in case something goes wrong during the upgrade. This could involve reverting to the old runtime (Edge 4.6.4) from LTS of 4.6.5 and redeploying the backup of your applications (exported jar file).

## Steps for CloudHub 2.0

1. **Navigate to Runtime Manager**
   * Go to Anypoint Platform > Runtime Manager.
2. **Select Application:**
   * Choose the application you want to rollback.
3. **Change Runtime Version:**
   * Go to the **Settings** tab of the application.
   * Under **Runtime**, change release channel to Edge and select version 4.6.4 from the dropdown.
   * Click Save and redeploy the application.

**1.Anypoint platform runtime manager (old version and release channel):**

A screenshot of a computer

Description automatically generated

2.**Anypoint platform runtime manager (New version and release channel):**

A screenshot of a computer

Description automatically generated

3.**Anypoint platform runtime manager in the application file, upload the backup or old exported jar file.**

A screenshot of a computer

Description automatically generated

4.**Checkin the old code or exported jar file into github repository.**

**5.Test Rollback Procedure:**

* + Test the rollback procedure in a dev/tst environment to ensure it can be done quickly and efficiently.
  + Implement the same steps for production rollback deployment.

Bottom of Form

# Mule Rollback Plan

## Prepare Rollback Plan

* + Have a rollback plan in place in case something goes wrong during the java version upgrade. This could involve reverting to the old java 8 from java 17 version and redeploying the backup of our applications (exported jar file).

## Steps for CloudHub 2.0

1. **Navigate to Runtime Manager**
   * Go to Anypoint Platform >>> Runtime Manager.
2. **Select Application:**
   * Choose the application we want to rollback.
3. **Change java Version:**
   * Go to the **Settings** tab of the application.
   * Under **Runtime deployment target**, rollback the java version 17 to 8.
   * Click **Save** and redeploy the application.

**1.Anypoint platform runtime manager (Current java version):**

A screenshot of a computer

Description automatically generated

2.**Anypoint platform runtime manager (Rollback java version):**

A screenshot of a computer

Description automatically generated

3.**Anypoint platform runtime manager in the application file, upload the backup or old exported jar file.**

A screenshot of a computer

Description automatically generated

**4.Anypoint studio:**

**Use the old jar file (the backup file) to roll back to the previous java 8 version dependencies and modules.**

**5.Check in the old code or exported jar file into github repository.**

**6.Test Rollback Procedure:**

* + Test the rollback procedure in a dev/tst environment to ensure it can be done quickly and efficiently.
  + Implement the same steps for production rollback deployment.