Hortonworks Data Platform

Apache Zeppelin Component Guide

(Nov 30, 2016)

docs.hortonworks.com

Hortonworks Data Platform: Apache Zeppelin Component Guide

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, Zookeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain free and open source.

Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to Contact Us directly to discuss your specific needs.

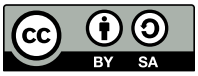
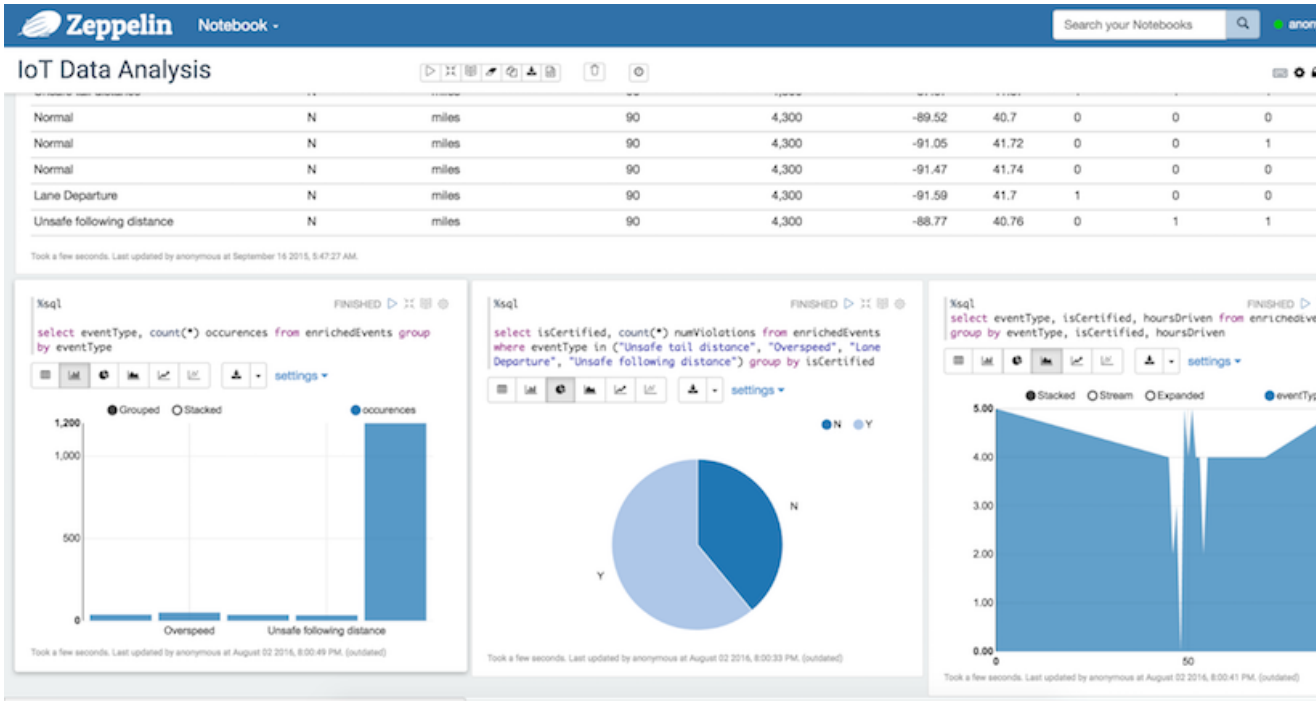


Table of Contents

1. Overview

Apache Zeppelin is a web-based notebook that supports interactive data exploration, visualization, sharing and collaboration. Zeppelin supports a growing list of programming languages and interfaces, including Python, Scala, Hive, SparkSQL, shell, AngularJS, and markdown.

Apache Zeppelin is useful for working interactively with long workflows: developing, organizing, running, and sharing analytic code and visualizing results.



2. Installing Apache Zeppelin

This chapter describes how to install Zeppelin on an Ambari-managed cluster. To install Zeppelin on a manually-managed HDP cluster, see Installing and Configuring Apache Zeppelin in the *Non-Ambari Cluster Installation Guide*.

To configure security for Zeppelin, see Configuring Zeppelin Security.

Note also that Livy is useful for accessing Spark from Zeppelin, providing security features and user impersonation support.

- On an Ambari-managed cluster, Ambari automatically installs Livy when you install Zeppelin.
- To install Livy on a manually-managed cluster, see "Installing and Configuring Livy" in the *Command Line Installation Guide*.

After installing Livy, see Configure the Livy Server for additional information.

2.1. Requirements for Installing Zeppelin

Install Zeppelin on a node where Spark clients are already installed and running. In addition, Java 8 is required on the node where Zeppelin is installed. This typically means that Zeppelin will be installed on a gateway or edge node.

Make sure your HDP cluster is running the following software:

- HDP 2.5.3 or later.
- Apache Spark version 1.6. Zeppelin does currently not support Spark 2.0, which is available as a technical preview with HDP 2.5.3.

2.2. Installing Zeppelin Using Ambari

This section describes how to install Zeppelin using Ambari. You will use the installation wizard to select Zeppelin and specify the node (or nodes) where Zeppelin will be installed.

The installation wizard sets useful default values for Zeppelin configuration settings. At first you should accept the default settings. Later, when you are more familiar with Zeppelin, you might decide to customize Zeppelin configuration settings.

To install Zeppelin using Ambari, add the Zeppelin service:

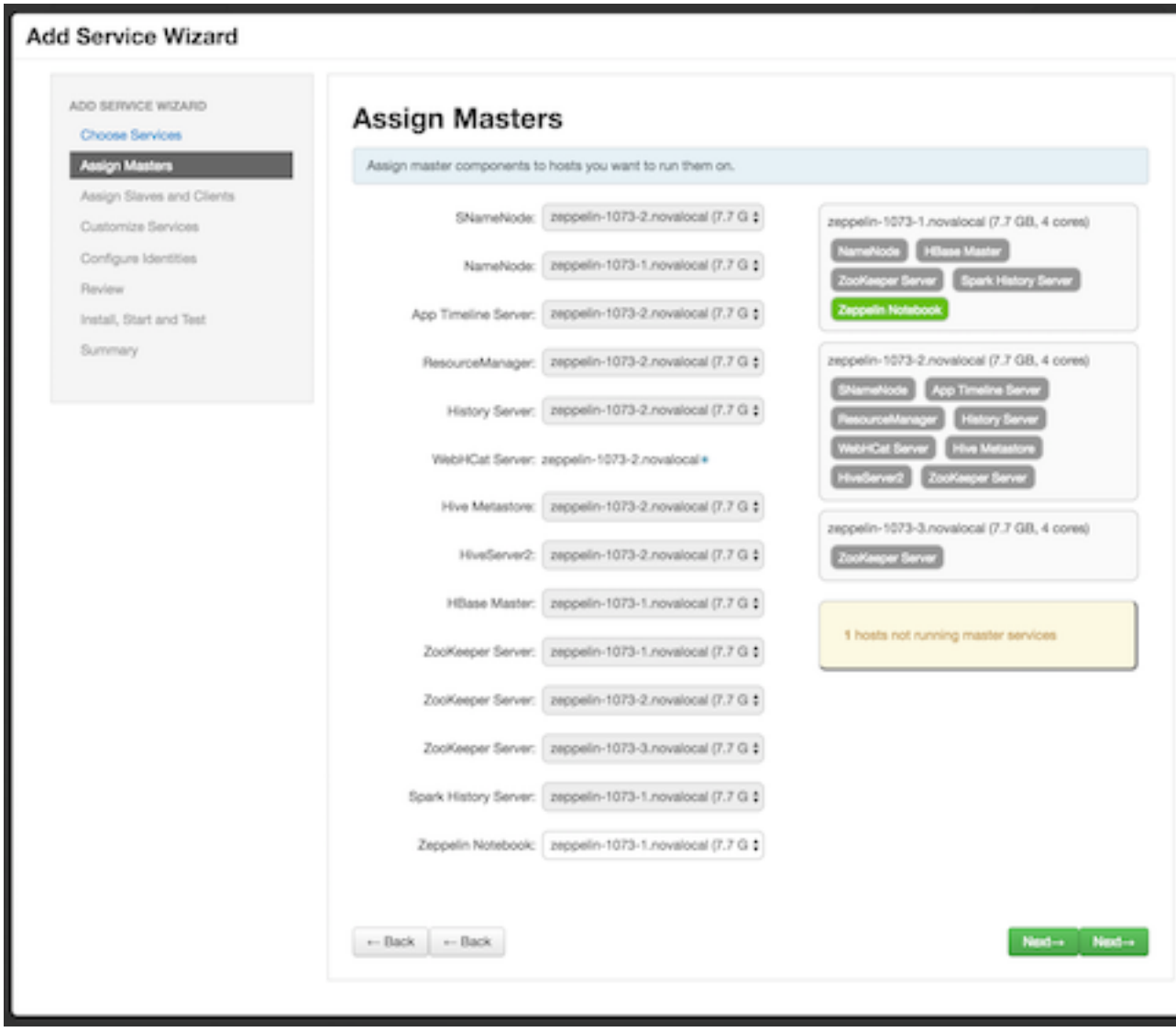
1. In the "Actions" menu at the bottom left of the Ambari dashboard, select "Add Service":

This starts the Add Service wizard, which displays the "Choose Services" page.

2. On the Choose Services page, select the Zeppelin service.

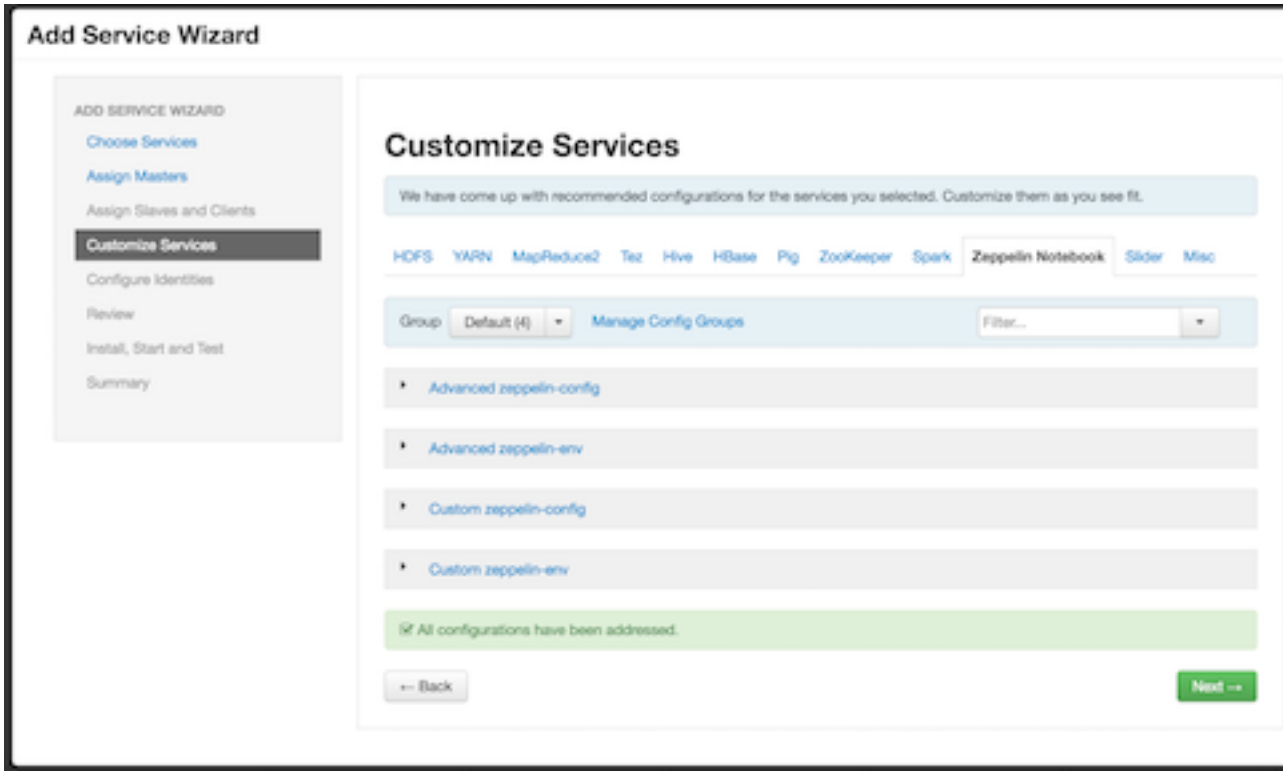
3. Click "Next" to continue.

4. On the Assign Masters page, review the node assignment for Zeppelin, and modify as needed:

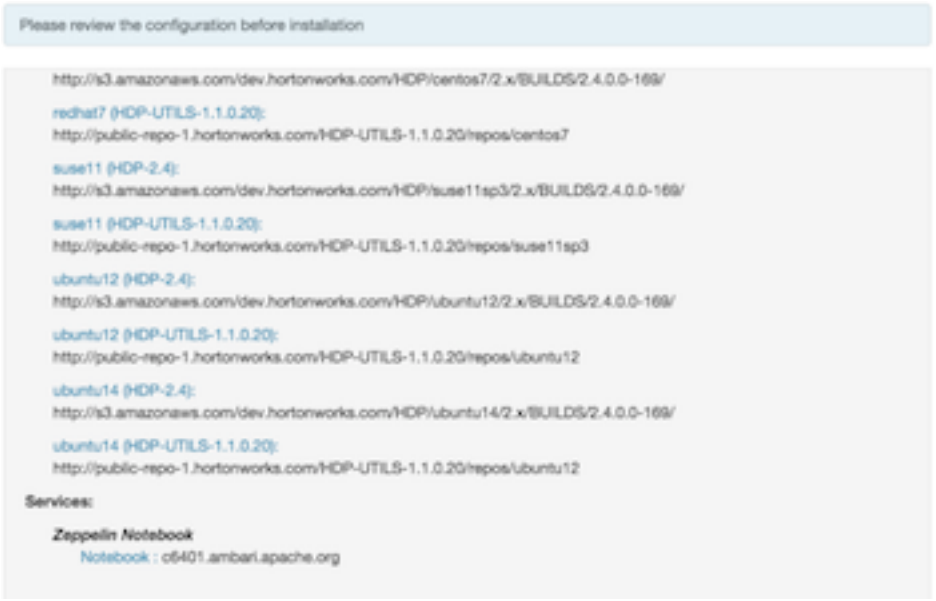


5. Click "Next" to continue.

6. On the Customize Services page, review default values and click "Next" to continue.



7. On the Review page, confirm the node selected to run the Zeppelin service:



8. Click "Deploy" to complete the installation process.

To validate your Zeppelin installation, open the Zeppelin hostname in a web browser. Use the port number configured for Zeppelin; for example:`http://<zeppelinhost>:9995`.

To check the version number of Zeppelin, type the following command on the command line:

```
/usr/hdp/current/zeppelin-server/bin/zeppelin-daemon.sh --version
```

Zeppelin stores configuration settings in the `/etc/zeppelin/conf` directory. Note, however, that if your cluster is managed by Ambari you should not modify configuration settings directly. Instead, use the Ambari web UI.

Zeppelin stores log files in `/var/log/zeppelin` on the node where Zeppelin is installed.

3. Using Zeppelin

A Zeppelin notebook is a browser-based GUI for interactive data exploration and modeling. As a notebook author or collaborator, you write code in a browser window. When you run the code from the browser, Zeppelin sends the code to backend processors such as Spark. The processor or service and returns results; you can then use Zeppelin to visualize results in the browser.

Zeppelin is structured around the following concepts, described in more detail in this chapter:

note	A note is the fundamental element of a Zeppelin notebook. Each note consists of one or more paragraphs of code.
paragraph	A paragraph contains code to access services, run jobs, and display results.
interpreter	An interpreter is a plugin that enables you to access processing engines and data sources from the Zeppelin UI. For example, if you want to use scala code in your Zeppelin notebook, you need a scala interpreter. Each interpreter runs in its own JVM on the same node as the Zeppelin server. The Zeppelin server communicates with interpreters through the use of Thrift.

Browser Support

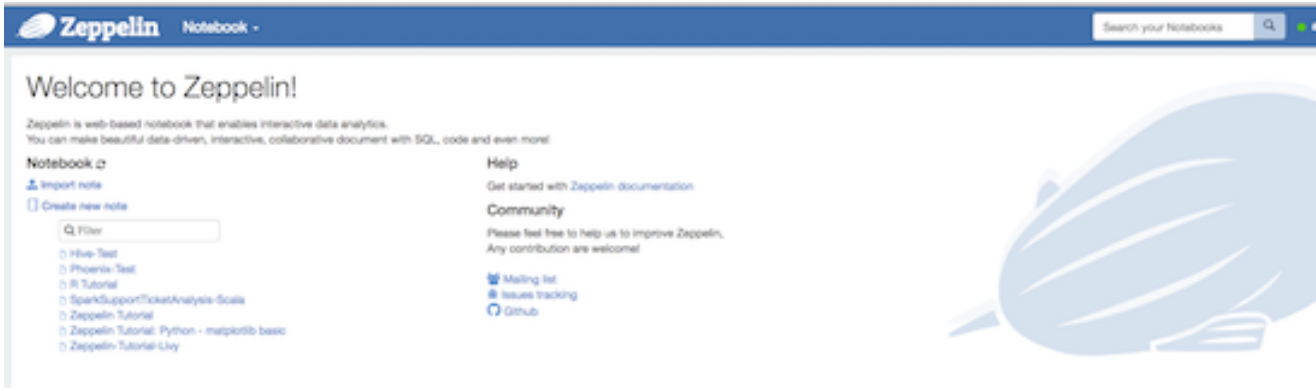
Zeppelin is supported on the following browsers:

- Internet Explorer versions 10 and 11. (Zeppelin is not supported on Internet Explorer version 8 or 9 due to lack of native support for WebSockets.)
- Google Chrome, latest stable release.
- Mozilla Firefox, latest stable release.
- Apple Safari, latest stable release. Note that when SSL is enabled for Zeppelin, Safari requires a Certificate Authority-signed certificate to access the Zeppelin UI.

3.1. Launching Zeppelin

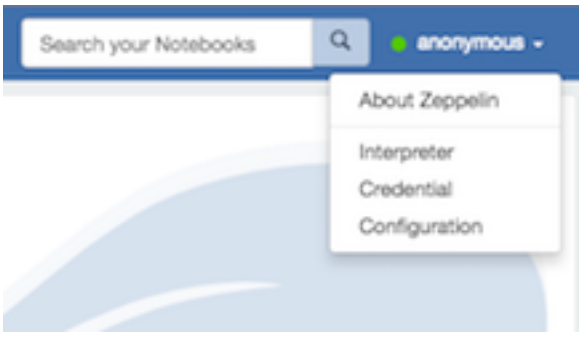
To launch Zeppelin in your browser, access the host and port associated with the Zeppelin server; for example, `http://<zeppelinhost>:9995`.

When you first connect to Zeppelin, you should see a welcome page similar to the following:



The following menus are available in the top banner of all Zeppelin pages:

- Notebook: Open a note, filter the list of notes by name, or create a note.
- Settings (listed as your username, or "anonymous" if you are using the default Shiro configuration):



  - List version information about Zeppelin
  - Review interpreter settings and configure, add, or remove interpreter instances
  - Save credentials for data sources
  - Display configuration settings

Zeppelin ships with sample notebooks, including a Zeppelin tutorial. Sample notebooks are stored in `/usr/hdp/current/zeppelin-server/notebook`. The tutorial demonstrates how to run Spark scala code, Spark SQL code, and built-in visualizations; and shows how to use form variables. To run the tutorial, click the Zeppelin tutorial link on the left side of the welcome page.

In addition to the Zeppelin Tutorial and other sample notebooks, there are sample notebooks available at https://github.com/hortonworks-gallery/zeppelin-notebooks.

3.2. Creating, Configuring, and Editing a Note

Each note consists of one or more paragraphs. Each paragraph consists of three main sections:

- An interactive box for code
- A box that displays results
- A paragraph toolbar

The following graphic shows all three sections of a paragraph:

To start work in Zeppelin, create a note: click "Create new note" on the welcome screen, or click on the "Notebook" drop-down menu and choose "Create new note."

When you save a new note, Zeppelin lists it on the left side of the welcome page. By default, notes are stored in the `$ZEPPELIN_HOME/notebook` folder.

Using the Note Toolbar

At the top of each note there is a toolbar with buttons for running the note and for setting configuration, security, and display options:

There are several buttons in the middle of the toolbar:

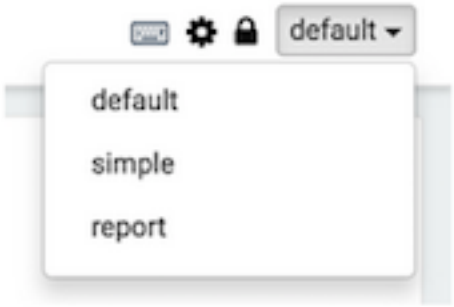
These buttons perform the following operations:

- Execute all paragraphs in the note sequentially, in the order in which they are displayed in the note.
- Hide or show the code section of all paragraphs.
- Hide or show the result sections in all paragraphs.
- Clear the result section in all paragraphs.
- Clone the current note.
- Export the current note to a JSON file.

Note that the code and result sections in all paragraphs are exported. If you have extra data in some of your result sections, trim the data before exporting it.

- Commit the current note content.
- Delete the note.
- Schedule the execution of all paragraphs using CRON syntax. This feature does not currently work, and it is not supported. If you need to schedule Spark jobs, look at Oozie Spark action.

There are additional buttons on the right side of the toolbar:



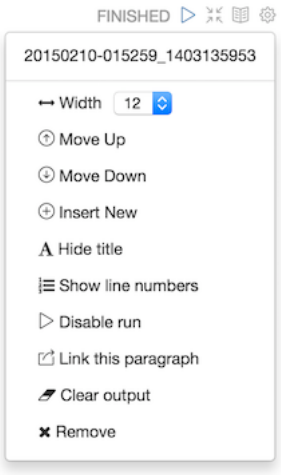
These buttons perform the following operations:

- Display all keyboard shortcuts.
- Configure interpreters that are bound to the current note.
- Configure note permissions.
- Switch display mode between default, simple, and report:
  - Default: the notebook can be shared with (and edited by) anyone who has access to the notebook.
  - Simple: similar to default, with available options shown only when your cursor is over the cell.
  - Report: only your results are visible, and are read-only (no editing).

Using the Paragraph Toolbar

The paragraph toolbar at the upper right of each paragraph provides buttons for running paragraph code, hiding or showing the code and result sections, and configuring the paragraph (for example, moving the paragraph up or down in the note, setting its width, or removing the paragraph from the note).

The settings icon (outlined in red) presents the following menu, which allows you to perform operations such as reordering paragraphs:



Running Code in Zeppelin

To run code in a note:

1. Type commands into a paragraph.
2. Click the triangle button.

   If you declared the associated interpreter strings at the start of your note, associated contexts are created automatically.
3. Results appear in the box underneath your code.

3.3. Configuring and Using Zeppelin Interpreters

A Zeppelin interpreter allows languages and data processing backends to be plugged into a Zeppelin notebook. Apache Zeppelin on HDP supports the following interpreters:

- Spark
- Hive (supported via JDBC)
- Shell
- MarkDown
- Livy (supports Spark, SparkSQL, PySpark, and SparkR)
- AngularJS

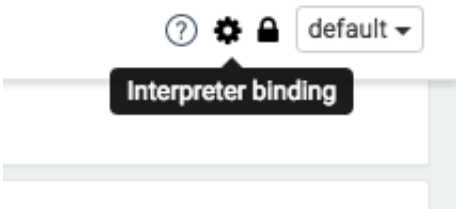
Configuring a Zeppelin Interpreter

This subsection describes how to select and configure interpreters for use in your notebook.

To select an interpreter:

1. Navigate to your notebook.

2. Click on “interpreter binding”:



3. On the interpreter page, selected interpreters appear in blue text; unselected interpreters appear in white text:

4. Review the list to ensure that any interpreters you wish to use are listed in blue. If not, click on the interpreter name to select the interpreter. Each click operates as a toggle.

5. When one or more interpreters could be used to access the same underlying service, you should deselect the interpreters that will not be used. This makes your choices clearer. For example, if you select `%livy`, unselect `%spark`.

6. When finished, click "Save".

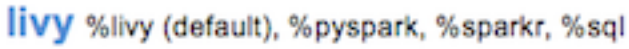
To set custom configuration values for an interpreter, scroll down to the Properties section for the interpreter and click "Edit". Edit and save the settings. For example, you might want to change the Spark home directory for the `spark` interpreter, set the name of the Hive JDBC driver for the JDBC interpreter, or set the keytab and principal name for a secure installation.

To specify precedence for the use of interpreters within a note, drag and drop interpreters into the desired positions in the list.

Using a Zeppelin Interpreter

To use an interpreter, add the associated interpreter directive with the format `%[INTERPRETER_NAME]` at the beginning of a paragraph in your note.

Each interpreter supports one or more directives. The list of `INTERPRETER_NAME` strings for an interpreter are at the start of the associated interpreter section of the notebook; for example:



You can find the information for an interpreter by scrolling through the list of interpreters, or by searching for the interpreter name. Interpreter entries list available directives, as well as interpreter options, property settings, and dependencies.

The following code specifies the default Spark interpreter:

```
%spark
```

The following code specifies the Livy interpreter:

```
%livy
```

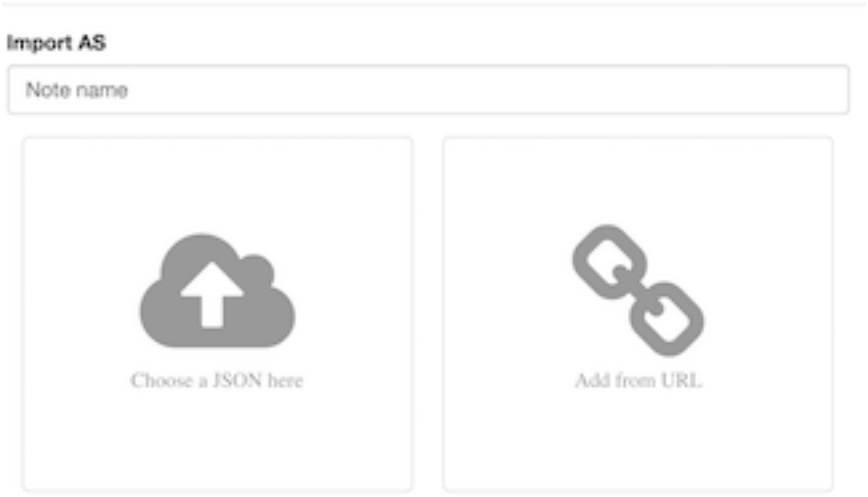
Each interpreter belongs to an interpreter group. Interpreters in the same group can reference each other. For example, when the Spark SQL interpreter and the Spark interpreter are in the same group, the Spark SQL interpreter can reference the Spark interpreter to access its SparkContext.

For more information about interpreters, see Interpreters in Apache Zeppelin at zeppelin.apache.org.

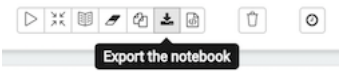
3.4. Importing and Exporting a Note

You can import notes that are available on a URL or on your local file system.

To import a note from a URL or from your local disk, click "Import note" on the welcome page, or use the drop-down menu to select "Import note":



To export a note, use the export note icon in the note toolbar:



Zeppelin downloads the note to the local file system as a JSON file.

3.5. Importing External Libraries

You might want to use one or more external libraries with Zeppelin. For example, the Magellan library requires that you import its dependencies.

To include an external dependency in a Zeppelin notebook, follow one of these approaches:

- Specify the dependency using an interpreter setting.
- Pass a jar, package, or file list to `spark-submit` using SPARK_SUBMIT_OPTIONS.

For more information, see Dependency Management for Interpreter in the Spark Interpreter for Apache Zeppelin documentation at zeppelin.apache.org.

If you want to import a library for a note that uses the Livy interpreter, see "Importing External Packages" in Using Zeppelin with Spark.

3.6. Using Zeppelin with Hive

To access Hive from Zeppelin, use the JDBC interpreter and specify Hive access:

```
%jdbc(hive)
```

The JDBC interpreter connects via Thrift.

To enable identity propagation with JDBC(Hive) interpreter:

1. Enable authentication via Shiro configuration, specifying authorization type, keytab, and principal:

   a. Set `zeppelin.jdbc.auth.type` to KERBEROS.

   b. Set `zeppelin.jdbc.principal` to the value of the principal.

   c. Set `zeppelin.jdbc.keytab.location` to the keytab location.

2. The JDBC interpreter automatically adds the end user as proxy user and sends the string to HiveServer2; for example:

```
jdbc:hive2://HiveHost:10000/default;principal=hive/_HOST@HOST1.COM;hive.
server2.proxy.user=testuser
```

To grant user `zeppelin` the ability to run in a Kerberos environment on an Ambari-managed cluster, add the following lines to the `hadoop.proxyuser.zeppelin.groups` property in the "Custom core-site" category of HDFS configuration settings.

For clusters not managed by Ambari, ensure that the following settings are in the `/etc/hadoop/conf/core-site.xml` file:

```
<property>
    <name>hadoop.proxyuser.zeppelin.groups</name>
    <value>*</value>
</property>
<property>
    <name>hadoop.proxyuser.zeppelin.hosts</name>
    <value>*</value>
</property>
```

If you want to enable user impersonation support during Hive query processing, perform one of the following steps:

- On an Ambari-managed cluster, navigate to the Hive configuration section and enable "Run as end user instead of Hive user."
- On a cluster not managed by Ambari, set `hive.server2.enable.doAs` to `true` in `/etc/hive2/conf/hive-default.xml`.

3.7. Using Zeppelin with Spark

To access Spark from Zeppelin, you can use the default interpreter for Spark:

```
%spark
```

To use an external library with the `%spark` interpreter, specify the dependency using a `%spark` interpreter setting (see Importing External Libraries for more information).

Note, however, that the default Spark interpreter has limitations that are addressed by an alternate interpreter based on Livy. For more efficient cluster resource utilization, to run jobs under the end user account (as opposed to the Zeppelin user), and to run Spark in yarn-cluster mode, you should access Spark through the Livy interpreter. The remainder of this subsection describes how to use Livy to access Spark.

Using Livy to Access Spark

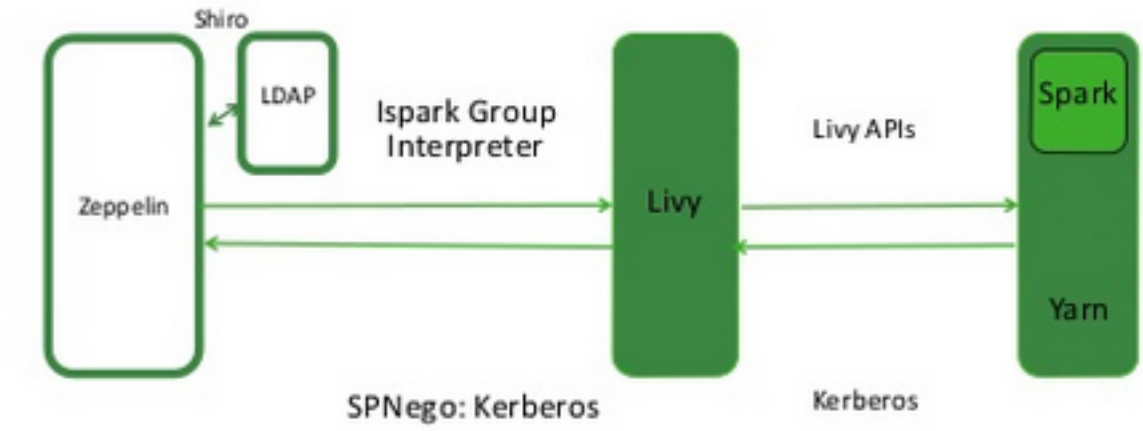
Livy is an open source REST interface for interacting with Spark. It runs code in a Spark context that runs locally or in YARN. To access Spark from Zeppelin through Livy, specify the Livy interpreter at the start of the paragraph that accesses Spark:

```
%livy
```

The Livy interpreter offers several advantages over the default Spark interpreter:

- Livy allows sharing of Spark context across multiple Zeppelin instances.
- Livy reduces resource use by recycling resources after 60 minutes of activity (by default), whereas the default Spark interpreter runs jobs–and retains job resources–indefinitely.
- Livy propagates user identity to the Spark job, so that the job runs as the originating user. The default Spark interpreter runs jobs as the default Zeppelin user.

The following graphic illustrates process interactions among Zeppelin, Livy, and Spark:



When the Zeppelin server runs with authentication enabled, the Livy interpreter uses Livy user impersonation. This is especially useful when multiple users are expected to connect to the same set of data repositories within an enterprise.

Using the Livy Interpreter

Before using Livy in a note, ensure that the Livy server is installed and running on your cluster, and that the Livy interpreter is configured properly for your cluster.

- For information about configuring the Livy server, see Configure the Livy Server.
- For general information about configuring and using Zeppelin interpreters, see Configuring and Using Zeppelin Interpreters.

To run code using Livy, specify the corresponding interpreter directive at the top of your note. For example, specify `%lspark` to run Scala code via Livy:

```
%lspark
sc.version
```

Important

To use SQLContext with Livy, do not create SQLContext explicitly. Zeppelin creates SQLContext by default.

If necessary, remove the following lines from the SparkSQL declaration area of your note:

```
//val sqlContext = new org.apache.spark.sql.SQLContext(sc)
//import sqlContext.implicits._
```

Livy sessions are recycled after a specified period of session inactivity. The default is one hour. If your session times out, you will need to restart the interpreter. On the Interpreters configuration page, locate the Livy interpreter and click "restart":

For information about specifying the session timeout value, see Configure the Livy Server.

Importing External Packages

To import an external package for use with a note that runs with Livy:

1. Navigate to the interpreter settings.
2. Edit the Livy interpreter setting as follows:
   a. Add a new key, `livy.spark.jars.packages`.
   b. Set its value to `group:id:version`. For example, for Databricks CSV package, set it to `com.databricks:spark-csv_2.11:1.4.0`.

4. Configuring Zeppelin Security

This chapter describes how to configure optional security features for Zeppelin:

- Configure Zeppelin for Authentication.
- Configure the Livy Server for interacting with Spark; Livy offers user impersonation and secure job submission.
- Enable access control for a Zeppelin notebook.
- Configure the Zeppelin UI for access over SSL (HTTPS).
- Configure Zeppelin for a Kerberos-enabled cluster.

4.1. Configure Zeppelin for Authentication

Zeppelin uses Apache Shiro to manage LDAP, active directory, basic authentication, and anonymous access. The following instructions describe how to configure Zeppelin for authentication. For additional information, see Shiro authentication for Apache Zeppelin.

To configure Zeppelin for authentication, follow these steps:

1. Configure basic authorization settings for Zeppelin.

   Using Ambari, access the `shiro_ini_content` property in the "Advanced zeppelin-env" section of Zeppelin configuration settings. (For clusters not managed by Ambari, the `shiro_ini_content` property corresponds to the contents of the Shiro configuration file, `/usr/hdp/current/zeppelin-server/lib/conf/shiro.ini`.)

   The property value consists of three blocks of attributes: `urls`, `users`, and `main`. Make sure that the following lines are in the `urls` section:

   ```
   [urls]
   /api/version = anon
   #/** = anon
   /** = authcBasic
   ```

2. Specify which users are allowed to access Zeppelin.

   To use Apache Shiro as the identity store, list authorized accounts and passwords in the `[users]` section of the `shiro_ini_content` property.

   For example, to enable authentication for users `admin`, `user1`, and `user2` with passwords `password1`, `password2`, and `password3`, respectively:

   ```
   [users]
   admin = password1
   user1 = password2
   user2 = password3
   ```

   Alternately, to use LDAP as the identity store, configure the `main` section for your LDAP settings. For example:

```
[main]
#ldapRealm = org.apache.shiro.realm.ldap.JndiLdapRealm
#ldapRealm.userDnTemplate = cn={0},cn=engg,ou=testdomain,dc=testdomain,dc=
com
#ldapRealm.contextFactory.url = ldap://ldaphost:389
#ldapRealm.contextFactory.authenticationMechanism = simple
```

3. Restart the Zeppelin server using Ambari or, on a non-Ambari managed cluster, follow the instructions in Installing and Configuring Apache Zeppelin in the *Non-Ambari Cluster Installation Guide*.

4. When using Zeppelin with authentication, you must first log on as a user authorized for Zeppelin.

4.2. Configure the Livy Server

Livy is an open source REST interface for interacting with Spark. Authorized users can launch a Spark session and submit code. Two different users can access their own private data and session, and they can collaborate on a notebook. Only the Livy server can submit a job securely to a Spark session.

Before using Livy, ensure that the Livy interpreter is configured properly for your cluster:

1. Review Livy interpreter settings:

2. Ensure that the default proxy permissions are correct. Restrict settings if necessary, but enable access only for groups and hosts where Livy is running.

   To grant user `livy` the ability to proxy users using Ambari, add the following lines to the `hadoop.proxyuser.livy.groups` property in the "Custom core-site" category of Zeppelin configuration settings.

   For clusters not managed by Ambari, edit the `/etc/hadoop/conf/core-site.xml` file:

```
<property>
    <name>hadoop.proxyuser.livy.groups</name>
    <value>*</value>
</property>

<property>
    <name>hadoop.proxyuser.livy.hosts</name>
    <value>*</value>
</property>
```

3. If Livy is installed on a different node than Zeppelin, replace `localhost` in the Livy URL with your Livy host.

4. To preserve cluster resources, Livy sessions are recycled after an hour of Livy session inactivity. When a Livy session times out, the Livy interpreter must be restarted.

   To specify a larger or smaller value using Ambari, configure the `livy.server.session.timeout` in the Advanced livy-conf section of the Spark service. Specify the timeout in milliseconds:

5. If you make any changes to the Livy interpreter settings, restart the Livy interpreter:

   On the Interpreters configuration page, locate the Livy interpreter and click "restart":

For information about configuring interpreters, see Configuring and Using Zeppelin Interpreters.

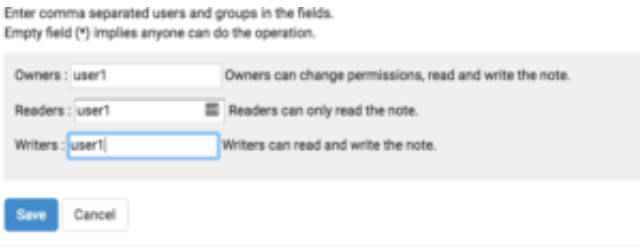
To use the Livy interpreter to access Spark, Spark SQL, PySpark or other services from a notebook, see Using Zeppelin with Spark.

4.3. Enable Access Control for a Zeppelin Notebook

To configure Zeppelin to authorize end users for Zeppelin notebook access:

1. Click the lock icon on the notebook to configure access to that notebook:

2. On the next popup menu, add users who should have access to the policy:

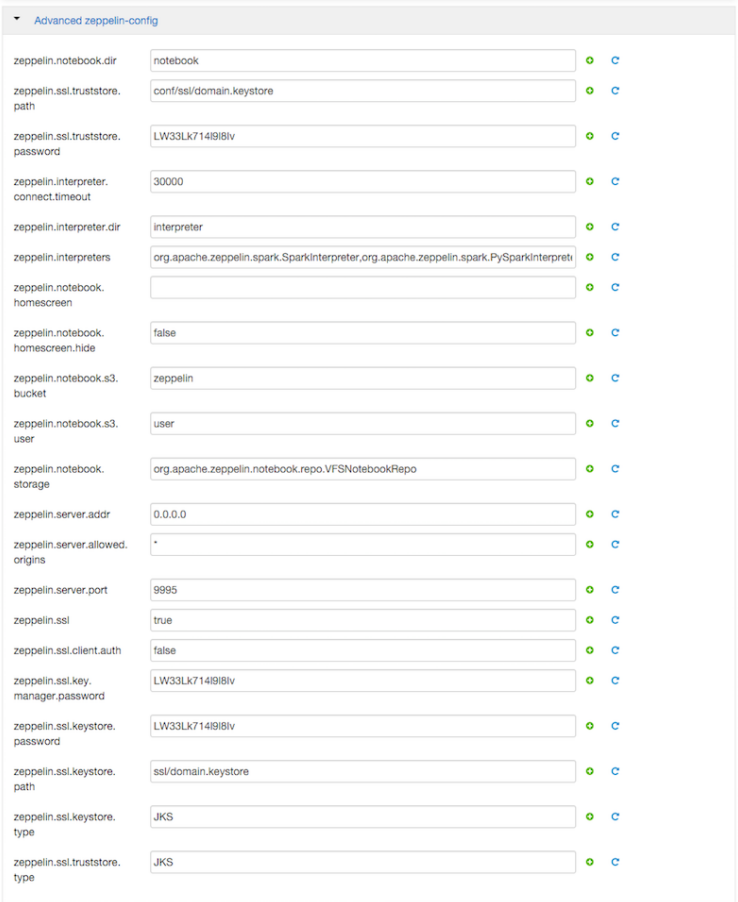


Note: When identity propagation is enabled via Livy, data access is controlled by the type of data source being accessed. For example, if you access HDFS, data access is controlled by HDFS permissions.

4.4. Configure SSL for Zeppelin

To configure SSL for Zeppelin using Ambari:

1. Access the "Advanced zeppelin-config" section of the Zeppelin configuration settings:



2. Set `zeppelin.ssl` to `true`.

3. Configure key manager and key store settings with the correct values for your system:

a. Set `zeppelin.ssl.key.manager.password` to the password associated with the key manager.

b. Set `zeppelin.ssl.keystore.password` to the password associated with the key store.

c. Set `zeppelin.ssl.keystore.path` to the path associated with the key store.

d. Set `zeppelin.ssl.keystore.type` to the type of key store configured on the cluster (for example, JKS).

4. If you wish to use client-side certificate authentication, enable client-side authentication and configure the associated trust store settings:

   a. Set `zeppelin.ssl.cient.auth` to `true`

   b. Set `zeppelin.ssl.truststore.path` to the path associated with your trust store.

   c. Set `zeppelin.ssl.truststore.password` to the password associated with your trust store.

   d. Set `zeppelin.ssl.truststore.type` to the type of trust store configured on the cluster (for example, JKS).

5. Check to make sure that all settings are valid.

6. Connect using HTTPS.

When SSL is enabled for Zeppelin, the Safari browser requires a Certificate Authority-signed certificate to access the Zeppelin UI.

4.5. Configure Zeppelin for a Kerberos-Enabled Cluster

The Zeppelin daemon needs a Kerberos account and keytab to run in a Kerberized cluster.

- When you enable Kerberos on an Ambari-managed cluster, Ambari configures Kerberos for Zeppelin and automatically creates a Kerberos account and keytab for it. For more information, see Configuring Ambari and Hadoop for Kerberos.
- If your cluster is not managed with Ambari and you plan to enable Kerberos for the Zeppelin server, see Creating Service Principals and Keytab Files for HDP in the *Hadoop Security Guide*.

5. Stopping the Zeppelin Server and Livy Server

To stop the Zeppelin server on a cluster managed by Ambari, use the Ambari Web UI.

To stop the Zeppelin server on a cluster that is not managed by Ambari, issue the following commands from user `zeppelin`:

```
/usr/hdp/current/zeppelin-server/bin/zeppelin-daemon.sh stop
```

To stop the Livy server:

```
su livy
cd /usr/hdp/current/livy-server
./bin/livy-server stop
```