# Aspire Introduction

> *Aspire is a framework and libraries of extensible components.*

Aspire is designed to enable creation of solutions to acquire data from one or more content repositories (such as file systems, relational databases, cloud storage, or content management systems).

- You can extract metadata and text from the documents.
- You can analyze, modify and enhance the content and metadata if needed.
- Then you can publish each document, together with its metadata, to a search engine or other target application.

> *Aspire uses Apache Felix (an open source implementation of OSGi).*

**On this page**



**Related pages**

- Prerequisites for Connectors and Content Processing
- Natural Language Processing (NLP)

## Why Aspire

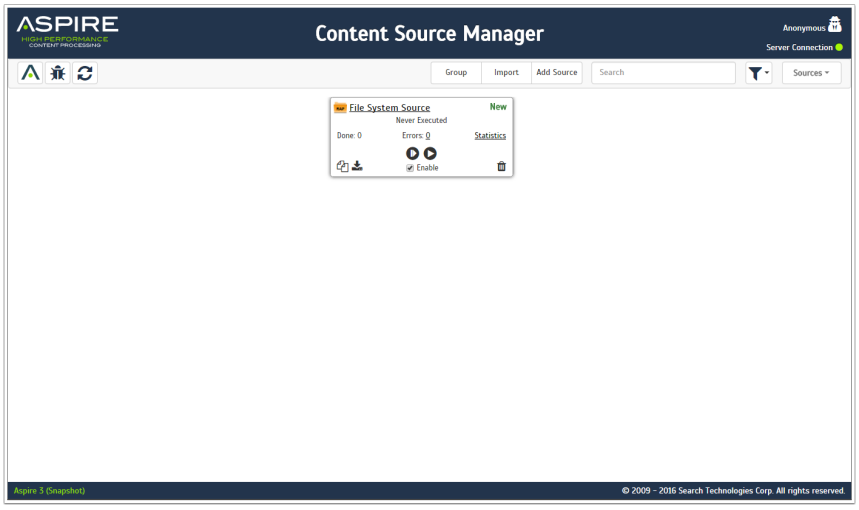
Aspire is flexible. By pulling data processing pipelines out of the search engine, Aspire uses its power and efficiency to:

- Manipulate content and metadata.
- Process that content and metadata in multiple pipelines simultaneously, and over multiple machines, for higher performance.
- Feed the content and metadata to one or more engines for indexing.

The Aspire framework supports creating Natural Language Processing (NLP), machine learning, and other analytic processing for text through a rich set of basic components.

You can install, start, stop, update, and uninstall Aspire components and applications.

- You won't need to reboot.
- You'll enjoy improved up-time and easier system administration.
- Each piece of Aspire processing functionality is a modular component that you can use by itself, or with other components to create an Aspire application.




## Administrator View

The administrator installs, configures, and maintains Aspire deployments. Manage Aspire deployments using a web-based, point-and-click interface, which is the same one an Aspire developer uses. However, an administrator needs to fill in only configuration information. Depending on your environment, you can use a single Aspire system administrator or several; each responsible for different content sources.

The System Administration UI has the following main functions.

- Configure properties for Aspire connectors
- Set up crawling schedules for repositories
- Manage full and incremental crawls
- Manage security
- Monitor system health and performance
- Monitor crawl statistics and performance
- Index auditing

**Administration UI security**

- Secure the Administration UI

**Connection security**

- Secure connections to Aspire

**Document level security**

- SOLR document security filtering
- Google Search Appliance filtering - Use any Aspire connector that provides the document ACLs is enough and normal GSA filtering works.
- SharePoint 2013 filtering (on premise)

# Developer View

Aspire deployments are built dynamically from components and sub-components.

- Aspire also includes the concept of *application bundles*, which are groups of pre-packaged components.
- These perform a specific function and contain embedded files to define their look and feel within the Aspire Administration UI.
- System developers can combine components easily to process data according to the specific needs of an application.
- You can mix Standard Aspire components with custom third-party components or with new components.

The high level developer's view of Aspire processing control is based on three major component types.

- Component managers
- Pipeline managers
- Tokenization manager

## Aspire Basic or Enterprise distribution

- Aspire Community (Deprecated)
- Aspire Basic and Enterprise

# Aspire Features

## Performance and reliability

- Distributed processing and automatic threading
- The ability to split document processing jobs into sub jobs that can run in parallel
- Standard technology to manage and restart processes on servers for high availability
- Can be placed within an architecture for backup failover

## Ease of administration

- Makes dynamic (on-the-fly) configuration changes
- Dynamically adds new components
- Dynamic refreshs of component code
- Rich built-in XML processing methods include XPath and XSLT
- Hierarchical component configuration
- Rich and comprehensive web-based administration and control interface

## Strong developer environment

- Intuitive workflow interface
- Supports processing content in diverse languages
- Easy mapping of document fields to search fields
- Rich built-in JSON and XML processing methods including XPath and XSLT
- Use of scripting to build complex processing components
- Hierarchical component configuration

- Tightly integrated with Maven repositories to share and load component code
- Process streams of tokens to  perform text analytics
    - Entity extraction
    - Latent semantic analysis
    - Document vector creation and comparison
    - Topic analysis

### Security support

- Handles Proxy LDAP requests including:
    - Authenticates users
    - Determines user group membership across a multitude of systems

### Federate search request support

- Distributed queries to multiple search engines
- Merged search results

### Hadoop support

- Writes to HDFS
- Includes Aspire within map / reduce jobs

## Structure of an Aspire Solution

You can divide Aspire deployments into three high-level functional areas: content access, content processing, and publishing.

- **Content access** - Fetches the documents and associated metadata from the content repositories.
    - The applications that perform this function are called Aspire Connectors, which support Application Programing Interfaces (APIs) of target repositories.
    - They access content, metadata, and security credentials.
    - Where available, Aspire connectors capture the full directory structure from the repository and support "browsable" enterprise site maps.

- **Content processing** - Analyses, augments, and transforms content.
    - Depending on the application, this can involve use of regular expressions or a wide range of complex semantic and statistical processing techniques.
    - Content processing can spawn Hadoop map / reduce jobs for larger processing tasks.

- **Publishing** - Components in an Aspire deployment push processed text from the content processing pipelines to a target system in the correct form and, where available, using the search engine's ingestion API.
    - The target system is typically a search engine or file directory.
    - The applications that perform this function are called Aspire Publishers.
    - XML and JSON output is also available.

## Functional Component Hierarchy

- **Component** - An atomic piece of Aspire logic.
- **Configurable Component** - A single component wrapped with Dynamic XML Forms (DXF) for use with the Admin UI.
- **Application or Application Bundle** - Multiple components wrapped with DFX, and optionally, configuration files.

## Version Numbering

Aspire Core releases use version numbers to identify which software an Aspire solution is built upon.

- Left most digit - Contains a major version that is reserved to denote the overall architecture.
- Second digit - Represents the minor version and denotes a release with new features.
- Third number - Optionally, represents the stability release version. This denotes a release with multiple bug fixes.
- Fourth digit - Optionally, represents a release a version with one or only a few bug fixes.

**Note:** Aspire Connector version numbers match the major and minor releases. Over time, the numbers after the major digit can diverge.

There is no version dependency between Aspire Core and Connectors / Publishers. This allows Search Technologies to release new versions (of Connectors or Publishers) independently of Aspire Core releases. Major version number must always match.