



CI (continuous integration)/CD (continuous delivery) in ColdFusion (2021 release)

WHITE PAPER

SAPPEY, BRIAN

Table of Contents

Introduction	3
ColdFusion 2018	4
Silent Installation ColdFusion 2018	4
Verifying the Silent Installation of ColdFusion 2018.....	7
Running the Silent Installer for ColdFusion 2021.....	8
Continuous Integration	10
Lifecycle and Definition	11
Version Control.....	12
Build	13
Testing.....	13
Deployment	14
Challenges.....	15
Benefits.....	15
ColdFusion 2021 CI/CD Tutorial	16
ColdFusion Installation Steps and Configuration.....	16
ColdFusion Package Manager and Code Scan	21
CFSetup	25
Adobe ColdFusion Docker Image	27
Creating REST Web Services with ColdFusion.....	27
ColdFusion Programmatic REST API Registration (Application.cfc)	32
Sample Components and Functions to build your REST APIs.....	33
Source Control.....	36
Continuous Integration Server	39
ColdFusion 2021 and Development/Staging/Production Servers	48
Continuous Integration and Testing.....	53
Performance Monitoring Toolset.....	54

Install the ColdFusion Performance Monitoring Toolset.....	54
Install the ColdFusion Performance Monitoring Toolset Silently.....	64
Silent Installer Properties.....	64
Scripting your pipeline to connect your PMT and ColdFusion	66
Continuous Delivery and Deployments.....	76
Continuous Delivery	76
Continuous Deployment.....	76
Closing Remarks.....	77

Introduction

In this white paper, you will have the opportunity to learn what ColdFusion 2021 has to offer when exploring a CI (continuous integration)/CD (continuous delivery) pipeline. One of the fundamental backbones of DevOps, a CI/CD pipeline will provide strategic advantages for your organization. It is important to note, this paper specifically covers the CI/CD process with respect to ColdFusion 2021, and will briefly touch on some possibilities with ColdFusion 2018. There are many tools and options that can be used when setting up a pipeline, a number of those tools will be reviewed throughout this paper.

ColdFusion 2021 has made it possible to move into the DevOps world with less resistance. The new release has some incredible developments and packed with new features that simplify many steps of the CI/CD pipeline.

This is yet another example of the product keeping pace, with dynamic industry changes. Advancements of ColdFusion have been pivotal to the continued growth of the language and have formed a cornerstone of a solid foundation for years to come.

CI/CD has become a mainstream software practice, and while ColdFusion has always had the ability to be plugged into a pipeline, the release of 2021 has made this task seamless. Adobe is pushing hard to compete, in the dynamic development space, with respect to DevOps, and of course, allowing ColdFusion Engineers to advance their careers.

As you read through this white paper you will begin to gather a sense of what a CI/CD pipeline will offer to your organization. You will soon discover no pipeline is a one size fits all, or simply put, not everything in this document may provide an answer for the exact way to implement a CI/CD pipeline.

There is no right or wrong on how your organization will cross the finish line. This journey you are about to begin will have some pitfalls, and traps along the way.

It is not realistic to think any organization can drop everything and instantly adopt principles of DevOps. This process is vast and can be complicated. Instead, focus on this progression as the chance to alleviate broken processes, revamp legacy code, create a more spontaneous development group, empower your team, and reduce the boring parts of coding and deployments.

There is no silver bullet for selecting the right tools or steps in implementing a CI/CD pipeline. As tempting as new technologies can be, they will not solve all your problems. If your current development processes are bad, throwing new technology into the mix will not correct existing issues. A CI/CD pipeline does not automatically erase your process issues. Rather, look at your team, and your technologies and pick options that will fit your business and development lifecycle needs. Let the process work for you.

If you need to start somewhere, go for CI (continuous integration). This will get you started on your path to build a sustainable, scaling, development pipeline for your organization. It would certainly

allow you to test the waters, and make sure the groundwork is laid for all future enhancements, development practices, and process changes. Let it serve to be the catalyst to catapult and advance your organization and team into modern development workflows.

ColdFusion 2018

Although ColdFusion 2021 is the focus of this paper, ColdFusion Developers around the world have been integrating into pipelines for some time. With the release of ColdFusion 2016, Adobe introduced its first ever, Docker Image. This major step meant that ColdFusion could now be containerized.

Cloud-based ColdFusion applications, modern methodologies, and the ability to automate development pipelines was now a reality. DevOps started to make sense, and the ability for developers to speed up development, secure their code and deploy in an automated step was a tremendous transformation for the language.

Advanced complex cloud architecture, microservices, and in general non monolithic apps could be deployed significantly easier with the introduction of ColdFusion 2018. Although we will jump into the discussion about ColdFusion 2021 further down, many of the topics we will cover can be implemented with ColdFusion 2018.

Let's quickly review and highlight a few of the features that ColdFusion 2018 has to offer with creating a CI pipeline. If you are currently running ColdFusion 2021, this section may be irrelevant to you based on your needs, if so, feel free to skip the following section, or use it as review before we launch into the ColdFusion 2021 tutorial.

Silent Installation ColdFusion 2018

Being able to create a CI pipeline starts with the ability to script. In ColdFusion 2018, you have the option to perform a silent installation. This means no interaction will be required with the user. By scripting your installation, you can automate the task of installing ColdFusion.

The silent installer will run on all platforms that ColdFusion 2018 currently supports. However, it will not automatically configure your web server after completion. The wsconfig tool will be required to complete the final steps when setting up your webserver.

To support the silent installer, a properties file will provide the necessary settings for the build. If you name the installer properties file `silent.properties` and save it in the same directory as the ColdFusion installation program, the installer will atomically consume the file. It should be noted that you can use any custom name or location you desire, just be sure to specify the filepath to have it execute in the installation. You may run the silent installer through the command line or a batch file/script.

To silently install the service:

Windows

- ColdFusion_2018_WWEJ_win64.exe -f silent.properties

Non-Windows

- ./coldfusion_installer.bin [-f propertiesFilePath]

Properties File Definitions

#Silent properties for ColdFusion 2018

INSTALLER_UI=SILENT

#Valid Values are full/trial/developer

SILENT_LICENSE_MODE=full

SILENT_SERIAL_NUMBER=

#Use when it is upgrade

SILENT_PREV_SERIAL_NUMBER=

#Valid values are ear/war/standalone

SILENT_INSTALLER_TYPE=standalone

SILENT_INSTALL_ODBC=true

SILENT_INSTALL_JNBRIDGE=true

#Remote administrator component for server remote start/stop

SILENT_INSTALL_ADMIN=true

SILENT_INSTALL_SOLR=true

SILENT_INSTALL_PDFG=true

#For Linux it would be as /opt/coldfusion2018 (Change it as per your path)

SILENT_INSTALL_FOLDER=C:\\ColdFusion2018

#Enable one of Prod(secure/prod)/dev profile. These 3 are mutually exclusive and only one of them can be true and others are false.

SILENT_ENABLE_PROD_SECURE_PROFILE=true

SILENT_ENABLE_PROD_DEFAULT_PROFILE=false

SILENT_ENABLE_DEV_PROFILE=false

#IP addresses from which Administrator can be accessed.

SILENT_ADMIN_IP=

#IP address of the CF installed machine through which PDFG/Solr service would be accessed remotely.

SILENT_JETTY_IP=

SILENT_ADMIN_USERNAME=admin

SILENT_ADMIN_PASSWORD=Adm1n\$12

SILENT_ENABLE_RDS=true

SILENT_RUNTIME_USER=

#Provide password when enable RDS is true

SILENT_RDS_PASSWORD=Adm1n\$12

#username/password of remote administrator component for server remote start/stop

SILENT_JETTY_USERNAME=admin

SILENT_JETTY_PASSWORD=Adm1n\$12

#Context Root for J2ee Installation

SILENT_CONTEXT_ROOT=

SILENT_AUTO_ENABLE_UPDATES=true

#For Linux it would be as /opt/coldfusion11 (Change it as per your path)

SILENT_PREV_CF_MIGR_DIR=C:\\ColdFusion11

#Enable/Disable the servlets depending on if they are not used by your application or not.

#Applicable only if production OR Production+Secure profile is enabled. For Dev profile, all are enabled by default.

ENABLE_RDS=false

ENABLE_WSRP=false

ENABLE_JSDEBUG=false

ENABLE_CFR=false

ENABLE_CFSWF=false

ENABLE_CFFORMS=false

Verifying the Silent Installation of ColdFusion 2018

When the installation of ColdFusion 2018 completes, you can easily verify the success by opening the generated log file. In the installation directory look for Adobe_ColdFusion_2018_Install* and verify. The destination of the log file was specified in the silent.properties earlier in the setup process.

The ColdFusion 2021 tutorial further down in this white paper will be significantly more detailed. Pieces of it will apply to ColdFusion 2018. If anything, review the tutorial and extract items that apply to ColdFusion 2018. Silent installations and script-based tasks are just a few items that can be leveraged in ColdFusion 2018 when embarking on your pipeline journey. Obviously, version control, testing, and monitoring are all applicable and can be done without the need of ColdFusion 2021 or out of the box CI tools.

One additional note, Adobe launched the Performance Monitoring Toolset (PMT) in the ColdFusion 2018 release. In the CD discussion, there will be insight about the possible use cases for the PMT in your pipeline.

Running the Silent Installer for ColdFusion 2021

To silently install the service, enter the following command:

Windows

- Execute *cfinstall.bat --file-name <file name> --installer-mode silent*
- Execute *cfinstall.bat -f <file name> -i silent*

Non-Windows

- Execute *./cfinstall.sh --file-name <file name> --installer-mode silent*
- Execute *./cfinstall.sh -f <file name> -i silent*

The file requires the below properties to be constructed in a specific format to install the service silently.

Windows Properties

Use it when upgrading from Standard to Enterprise OR from ColdFusion 2018 to ColdFusion 2021
PREVIOUS_SERIAL_NUMBER=

ColdFusion Administrator password
COLDFUSION_ADMIN_PASSWORD=Adm1n\$123

Internal webserver port
COLDFUSION_ADMIN_PORT=8500

Install type - 1-Install new version of Adobe ColdFusion 2021 with a serial number, 2-30-day trial,
3-Developer Edition
INSTALL_TYPE=1

Accept EULA- true. To proceed, you must accept the EULA
EULA_ACCEPTED=true

Enter the serial number

SERIAL_NUMBER=

IP address from which Administrator can be accessed

IP_ADDRESSES=

Specify the deployment type- Production, Development, Staging, Testing, Disaster recovery

DEPLOYMENT_TYPE=Production

Name of the ColdFusion service

COLDFUSION_WIN_SERVICE_NAME=cf2021

Server profile - 1-Production+Secure, 2-Production, 3-Development.

SERVER_PROFILE=1

Specify the RDS password

COLDFUSION_RDS_PASSWORD=Adm1n\$123

ColdFusion install location

COLDFUSION_INSTALL_LOCATION=C:\ColdFusion_2021_WWEJ_win64\ColdFusion\

Non-Windows Properties

The serial number of the previous version of ColdFusion. Use it when it is for upgrade

PREVIOUS_SERIAL_NUMBER=

ColdFusion Administrator password

COLDFUSION_ADMIN_PASSWORD=Adm1n\$123

Internal webserver port

COLDFUSION_ADMIN_PORT=8500

ColdFusion runtime user. Applicable for Linux and Solaris

COLDFUSION_RUNTIME_USER=

Install type - 1-Install new version of Adobe ColdFusion 2021 with a serial number, 2-30-day trial, 3-Developer Edition

INSTALL_TYPE=1

Accept EULA- true. To proceed, you must accept the EULA

EULA_ACCEPTED=true

Serial number of the new version of ColdFusion

SERIAL_NUMBER=

IP address from which Administrator can be accessed
IP_ADDRESSES=

Specify the deployment type- Production, Development, Staging, Testing, Disaster recovery
DEPLOYMENT_TYPE=

Server profile - 1-Production+Secure, 2-Production, 3-Development.
SERVER_PROFILE=1

Specify the RDS password
COLDFUSION_RDS_PASSWORD=Adm1n\$123

ColdFusion install location
COLDFUSION_INSTALL_LOCATION=

File Paths

Windows

COLDFUSION_INSTALL_LOCATION= C:\ColdFusion_2021_WWEJ_win64\ColdFusion\

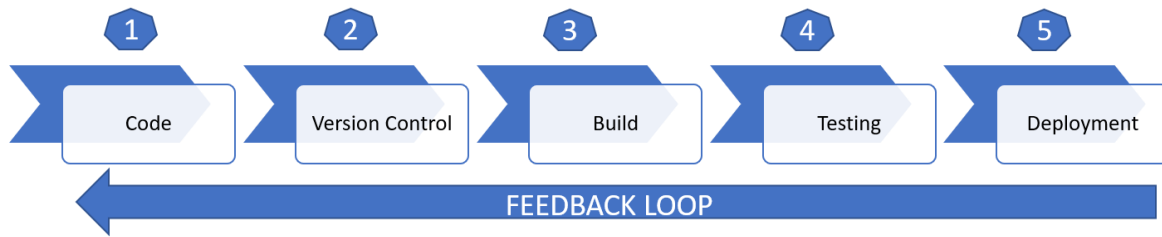
Linux and Solaris

COLDFUSION_INSTALL_LOCATION=/opt/ColdFusion/

Mac OS X

COLDFUSION_INSTALL_LOCATION=/Applications/ColdFusion/

Continuous Integration



Lifecycle and Definition

We will be using acronyms throughout this article. Let's quickly cover the CI (continuous integration) part of the pipeline.

This concept can take the form of many shapes, but overall, the focus is straightforward. Providing autonomy for your teams to code, build, test and automate deployments. For the ColdFusion world, this may present some challenges.

Let's be honest, how many times have you heard the following when it comes to a conversation about CI?

"it will never work here; we do things differently"

"it will slow us down; it doesn't make any difference"

What many professionals discover is that CI is not that hard. To develop and maintain a pipeline, with the right implementation and tools, significant improvements will be obvious immediately. The long-term benefits will outweigh any struggles with moving to a CI workflow.

Coding, testing and deployment processes can vary from team to team. Please keep the focus on the conceptual parts of what this paper is highlighting. Make this work for you. Keep the rules light and simple.

Let's review the different parts of a pipeline and what ColdFusion 2021 has to offer, to make your pipeline a fully functional part of your team's DevOps backbone.

Code

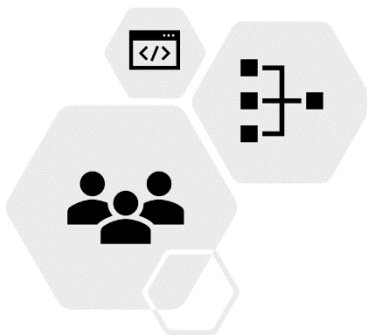


A solid use case would be the development of a REST API. By their nature, they are typically stand-alone applications that can be deployed independently and have potentially less impact than larger, monolithic applications. These applications are loosely coupled and may require many changes frequently, to meet the demands of the business, and to keep customers engaged and happy.

In development, REST API's play a critical role in websites, phone apps, and various other computing programs. Planning and writing the code can produce everything from simple GET's to complex POST's, in any fashion. ColdFusion is perfectly capable of building modern and highly scalable solutions. So why not push these applications through a pipeline?

Utilizing the ZIP Installer, or the Adobe Docker Image, that ships natively with ColdFusion 2021, you can have a self-contained ColdFusion server running on your local machine in a matter of minutes. Allowing you to quickly code a feature and commit that code to a repository (version control).

Version Control

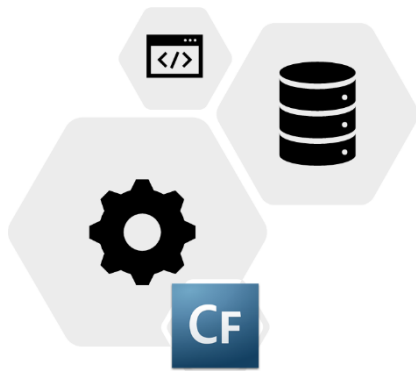


No matter the size of your team, or size of the project, a version control system is an integral part of any pipeline. It does not matter if you are building web applications, REST APIs, or phone applications, version control is the tool to connect the development process and deployments.

In a CI/CD pipeline, developers will be committing code to repositories multiple times a day. Keeping track and monitoring code changes with respect to your build is a critical component in keeping your builds healthy and environments in sync. With any well-defined workflow, catching bugs early in the process is a staple of the pipeline.

A great use case to demonstrate the importance of version control is determining what broke your last build in the CI feedback loop. Items like this always need to be carefully managed and monitored. Without some type of source control, and tagging, this task would be extremely painful and time consuming. Once those code commits are pushed, the build process must begin.

Build



Moving source code from local development environments can be a complicated process. Merging files, adding new configurations, updating assets etc., can present serious risks to an organization. This is where the automation can assist. Turning these steps into repeatable, automated tasks, will reduce mistakes, and avoid wasting time with manual builds.

ColdFusion 2021 provides several new enhancements which will make the build process even easier. With the introduction of command line features, and the ability to build and launch

systems using scripts, it should alleviate the challenges encountered in previous versions. 2021 will simplify your build process, even without a CI server running behind the scenes.

These command line features can be customized to your specific needs. Your build process should allow you to checkout code from a repository, and fire off any scripts required to execute the build.

A well-defined build process should not be tricky. Properly configured scripts should be able to analyze your code and only deploy what is needed. This would hold especially true for a large codebase. Instead of having to deploy the entire project, a build process should be able to deploy only the deltas and execute it quickly. This is another area where 2021 will be able to assist. With a new scan utility, that is script based, ColdFusion can inspect the code for required packages and only deploy what is needed.

Testing



Continuous testing is the process of testing an application continuously throughout the development cycle. There are numerous advantages of testing early in your pipeline. In most cases, these types of tests are significantly earlier in the process than traditional types of testing. Providing feedback to developers immediately when test cases begin to fail is a critical part of the pipeline and its effectiveness. A little later in this paper a concept of shift-left testing will be discussed. It is worth noting here since it can help to solve several testing failures, that are more common in a traditional sense of

testing.

Scaling a large organization and increasing test coverage is vital for growth. As your code and development team increases, so do the chances for mistakes, and not to mention the complexity of changing business rules.

Manual testing simply does not scale. It is not feasible to think humans can keep pace with large codebases and maintain constant coverage of all daily commits. Automated testing allows for scale, which is crucial for your blossoming pipeline.

Speed plays a critical role in this phase. The scaling effort of removing manual testing allows for incredible velocities to be achieved during the testing phase. This ultimately increases the application development rates and significantly reduces errors that can trickle up throughout the process. The faster the bugs are found, the easier it is to fix them. Bug dependencies get significantly more complicated to fix once the code goes to production. The longer that code bakes, the complexity of fixing these bugs greatly increases.

Deployment



Once the testing phase has been completed you are now ready for the first deployment. At this stage, you will be automating a push to your test or staging servers.

This again, is an area where ColdFusion 2021 has tremendous value. From the command line you can execute several scripts to assist in deploying the code, scan the deployment for missing dependencies, test the code, and ensure that the build is automated in a repeated fashion, time and time again.

The human element of manually pushing this code is removed. You can ensure that deployment mistakes and resource hours are not wasted waiting for the deployment to finish. Rather, your team is focusing on the feedback loop from your pipeline. Ready to monitor and fix any issues that may appear because of the deployment.

Depending on your organizational needs, many different things could be required in a deployment. In some cases, test code could be part of the deployment process. A good example would be an API. It would be advantageous to include a unit test with the deployment that could be quickly executed as a pulse check. This would allow for constant health checks against the build and that the code was successfully deployed and is executable.

You should leave yourself the flexibility to include options into the deployment scripts. In some cases, you may find, a simple deployment of code may be the only necessary item for the target process. In other cases, health checks, unit tests, or even simulators may be a required step.

One quick thing to mention. Some IDE's allow for build processes. They are proprietary tools that dictate the build. This may initially appear to simplify the process; but it could lock you in to the tools you select. It is prudent to avoid locking yourself into specific tools, and third-party dependencies when performing your builds and deployments.

Challenges

Some of the largest hurdles you will face will be implementing change in an organization where the employees and management will resist. To be clear, ColdFusion is just as capable as any other language when implementing a pipeline for your applications. If you have put off moving to a pipeline, your wait is not in vain, ColdFusion 2021 will be the perfect release to begin your exploration.

Shifting to a CI/CD workflow will require changes in your existing processes. It would be foolish to expect that these changes will not raise the possibility of potential risks. The key will be to gain support with small wins. Determine parts in your system that could be potential candidates, and tackle those first. Reinforce that with the release of ColdFusion 2021, and your new pipeline journey should be achievable.

The tutorials presented later in this paper will provide steps that can be used as an example to get the foundational pieces of the pipeline in place. Prior to this ColdFusion release, many features and staples of the CI/CD process would have to be custom built. With 2021, the command line features, code utilities, and package manager will allow scripting and automation directly in your CI pipeline, without any headache.

Benefits

The focus so far has highlighted the technical advantages of a CI/CD pipeline. Technical advancements are just one aspect in your organizations scope of potential benefits. An organization that is highly dependent on their end user's satisfaction can profit greatly from the CI/CD process. Those first impressions will greatly impact return visits. Keeping your product up to date and modern, fresh with new features, will promote customer satisfaction.

The recipe for this success is deep in the process. CI/CD promotes small and frequent integrations. Since they are typically smaller, and recent, the code remains in context for the developer. This alleviates mistakes and problems when sitting on massive code updates for weeks or months at a time. Not to mention, the immediate testing and feedback provided to the developer allows for an easier process to immediately address issues before too much time passes by.

Problems are detected early and fixed before new project code is integrated into the repository, possibly creating an even bigger problem that becomes increasingly more complex. With more development teams working from home, and increasing in size and intricacy, this feedback loop is critical for the health of an organization.

CI/CD continuously merges code and executes test cases. This provides a code base that is in a release ready state. This is another spot where ColdFusion 2021 can assist. With the ability to spin up a server via command line, and preconfigure options, developers are building code against an environment that closely mimics production. How liberating, allowing developers to worry about code and not environments, or even worse, deployments and bugs, rather than innovating and coding.

It is slightly out of the scope of this article, but Containerization is a great option to assist with these type functions. From building development containers to production containers, it plays quite nicely with CI/CD.

ColdFusion 2021 CI/CD Tutorial

As mentioned earlier, there are many different tools that you can apply when building out a pipeline with ColdFusion 2021 as your application development language.

For the tutorial, we will be using Git for source control along with ColdFusion 2021. Jenkins will be the CI server, but there is no hard and fast rule that Jenkins must be your primary choice when designing and implementing your pipeline.

Any CI server can be selected, and a CI server is not even required. There are many different options when selecting the configurations. The marketplace of tools is immense, a few will be mentioned throughout the tutorial, so you can gain context of the choices you could exercise. It is important to note that irrespective of Jenkins or any other CI server and tools, the process, procedures, and steps outlined below will stay the same.

ColdFusion Installation Steps and Configuration

This is a critical step in the pipeline. We will be reviewing the entire installation process for all parts that are required for the CI/CD pipeline.

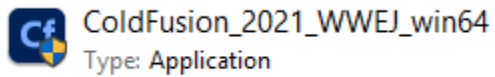
In this case, the development server you will configure will be done through a command line setup. This is a repeatable step that you can use to deploy your code and server configurations to your integration servers, staging/testing, and production environments.

The ColdFusion setup will go step by step, but these can be configured as arguments with a single command line option (CFSetup). This will allow your team the ability to store the config and execute the build for any server, where you need ColdFusion to run as a service. This also applies to running it local for development.

Although the tutorial will go over a command line utility setup with a step by step command line prompt, it is important to discuss one additional feature.

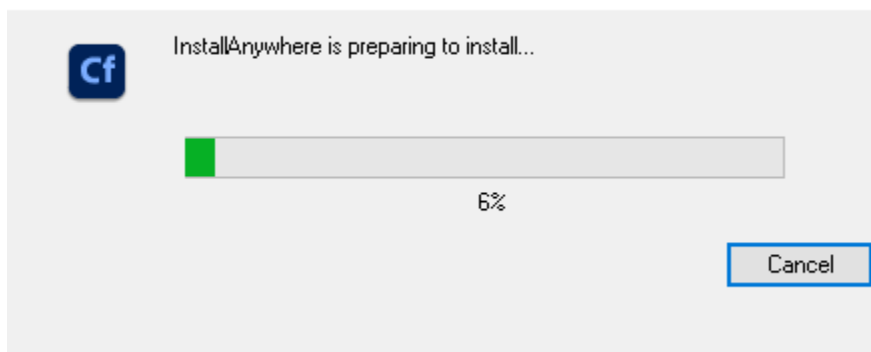
Download and Command Line Installation

Step 1: Download a copy of ColdFusion 2021 by visiting <https://www.adobe.com/support/coldfusion/downloads.html#cf2021productdownloads>



Step 2: If you do prefer the GUI installation begin the installation of ColdFusion for your selected OS. If you prefer the command line installation option after the download, please skip to **Step 3**

InstallAnywhere



Step 3: Windows Installation –

- a) Navigate to ColdFusion/cfusion/bin.
- b) Enter cfinstall.bat and follow the on-screen instructions.

```
Directory of C:\ColdFusion2021\cfusion\bin

02/18/2021  02:55 PM    <DIR>          .
02/18/2021  02:55 PM    <DIR>          ..
10/26/2020  07:32 PM             13,370 cf-bootstrap.jar
10/26/2020  07:32 PM             8,918 cf-lambdazip.jar
10/26/2020  07:32 PM            16,611 cf-osgicli.jar
10/26/2020  07:32 PM             4,682 cf-passwordreset.jar
10/26/2020  07:32 PM            79,217 cf-startup.jar
10/26/2020  07:32 PM             2,776 cf.bat
10/26/2020  07:32 PM             495 cfchart_xmltojson.bat
10/26/2020  07:32 PM             5,043 cfcompile.bat
10/26/2020  07:32 PM          1,501,420 cfencode.exe
10/26/2020  07:32 PM             476 cfinfo.bat
10/26/2020  07:32 PM             149 cfinstall.bat
10/26/2020  07:32 PM          2,265,164 cfinstall.jar
02/10/2021  09:47 PM              72 cfinstall.log
10/26/2020  07:32 PM             510 cfpm.bat
10/26/2020  07:32 PM            1,134 cfscan.bat
10/26/2020  07:32 PM             207 cfstart.bat
10/26/2020  07:32 PM             266 cfstat.bat
10/26/2020  07:32 PM             130 cfstop.bat
10/26/2020  07:32 PM             104 cfuninstall.bat
10/26/2020  07:32 PM          637,440 coldfusion.exe
10/26/2020  07:32 PM          75,776 coldfusionsvc.exe
10/26/2020  07:32 PM           5,120 coldfusionsvcmsg.dll
```

Step 4: Select an installation directory

cmd C:\WINDOWS\System32\cmd.exe

```
-----  
Welcome to the ColdFusion Setup Wizard  
-----  
=====
```

ColdFusion Installation Directory

Select the directory where Adobe ColdFusion 2021 is installed.
ColdFusion Installation Directory [C:\ColdFusion2021]: _

Step 5: It is important to note. To start the ColdFusion server it is mandatory to accept the EULA agreement.

```
=====
```

License Agreement

Installation and use of Adobe ColdFusion 2021 requires acceptance of the License Agreement.
license.txt>)

DO YOU ACCEPT THE TERMS OF LICENSE AGREEMENT? [Y/N]:

Step 6: Select your install type. For this tutorial the 30-day trial will be selected.

```
=====
```

Install Type

If you do not have a serial number, select either 30-day trial or Developer Edition.
For further details, visit http://www.adobe.com/go/cf_installtype.

1. Install new version of Adobe ColdFusion 2021 with a serial number
2. 30-day trial
3. Developer Edition

Installation Choice [1]:

Step 7: Select your server profile. For this tutorial the Development Profile will be used.


```
=====
Select ColdFusion Server Profile
-----

Development Profile: Use this profile only for development purposes. Note that features like Server Debugging and RDS are enabled for this profile.

Production Profile: Use this profile for production purposes. All debug features and RDS are disabled for this profile.

Production Profile + Secure Profile: Use this profile for a highly-secure production deployment that will allow a more fine-grained control over security. For details, see the secure profile guide (http://www.adobe.com/go/cf\_secureprofile).

Allowed admin IPs are the client IP addresses that can access the ColdFusion administrator. They can be a comma separated list of IP addresses (e.g., 10.0.0.1, 11.180.26.32, etc.). IP addresses can range from 10-30, or * wild cards. Both IPv4 and IPv6 addresses are supported.

When the installation completes, please lock down your Server as per the guidelines provided in the ColdFusion Lockdown Guide (http://www.adobe.com/go/cf\_lockdown).

1. Production Profile + Secure Profile
2. Production Profile
3. Development Profile

Select ColdFusion Server Profile [1]: _
```

Step 8: Enter an Administrator Password

```
=====
Administrator Password
-----

Enter the password that you will use to restrict access to the ColdFusion Administrator.
The password must contain at least 1 lower case, 1 numeric, and 1 special character (~!$%^&*()_-=,./;[]{}| -@#).

Enter Administrator Password:
```

Step 9: Select your port number. 8500 will work for local development.

```
=====
Built-in Web Server Port Number
-----

The Adobe ColdFusion 2021 Built-in Web Server will be configured to use port 8500.

ColdFusion Administrator Port [8500]: _
```

Step 10: RDS Password selection

```
=====
RDS Password
-----

The ColdFusion Remote Development Service (RDS) lets developers use Adobe tools remotely to connect to this server for development purposes.
Debugging and Remote Start/Stop from ColdFusion Builder, Line Debugging, Report Builder, and Dreamweaver Extensions.
Note, however, that disabling RDS also disables debugging, Security Analyzer.
It is recommended to have a password with at least 8 characters, 1 letter in upper case, 1 letter in lower case, 1 numeric, and 1 special character (~!$%^&*()_-=,./;[]{}| -@#).

Enter RDS Password: *****_
```

Step 11: Create a service name

```
=====
Adobe ColdFusion Service
-----

Service Name for ColdFusion Application Server [ColdFusion 2021 Application Server]:
```

Step 12: Select your deployment type

```
=====
Deployment Type
-----

Specify the type of deployment. Choose from the options below:

1. Production
2. Development
3. Staging
4. Testing
5. Disaster recovery

Deployment Type [1]:
```

Step 13: Your ColdFusion 2021 server should now be up and running. We will cover some additional items in the next section.

ColdFusion Package Manager and Code Scan

This is one of the most exciting parts of the new ColdFusion release. The concept of modularization has been around for many years in software development. The ability to divide systems into discrete and independent modules is extremely powerful, and the Adobe team has come through with an amazing enhancement.

The benefits are vast, but a brief overview and highlight of the benefits should provide some valuable insight into how this can assist when creating and developing your pipeline.

In previous versions of ColdFusion, you had a few options to pick and choose the components that were installed by default. This meant packages like datasources, and charting were automatically included in the build. In ColdFusion 2018 there were only two packages that could be left out of the install, PDFg and Solr.

Having extra packages that were not used only lead to a strain on system resources. Even worse, there are plenty of cases where these system drains were not required. A great example would be a REST API.

Most likely, if you are building standalone APIs that serve a single purpose, many of the components of ColdFusion that would be running in your API environment would be completely useless. Leading to bloat and potential overhead to a system that needs to be lean and fast.

ColdFusion 2021 is a game changer. You can now avoid unnecessary overhead and customize your package options based on your needs.

For example, you may want to execute Oracle database operations in your code. You simply install the Oracle package, and that alone. Extra packages bloating your build are not required and would not be included. If for some reason your needs change, you can uninstall the package. This is a tremendous advantage when building modern applications.

What does this mean? You now have a lightweight base version of ColdFusion and depending on your use, you can add or remove the packages on demand. If this wasn't already amazing, a ColdFusion restart is not required for these changes to take effect. This is a breakthrough for the language, and one of the most crucial enhancements to keep ColdFusion relevant in a modern development environment.

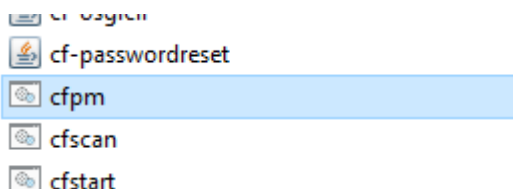
Before we get into the command line features, here are several key benefits:

1. The size of the installer now stands at 147 MB for the core ColdFusion services, which also includes the JRE.
2. The size on disk is now reduced by ~300% and the runtime memory (for core modules) is now reduced by ~350%. The disk size may increase once you install the other modules.
3. The startup time has decreased by ~400%.

Let's look at different options and how this could fit into a pipeline. Be sure that ColdFusion Status is actively running, and you have access to the command line for your service.

Step 1: Navigate to CFHOME/cfusion/bin and enter cfpm.bat

- a) After you execute the .bat you will have a command line prompt appear.



Option 1 (List Command): There are many different options you can run on the command line when your prompt is ready. We will cover just a few to provide some use cases.

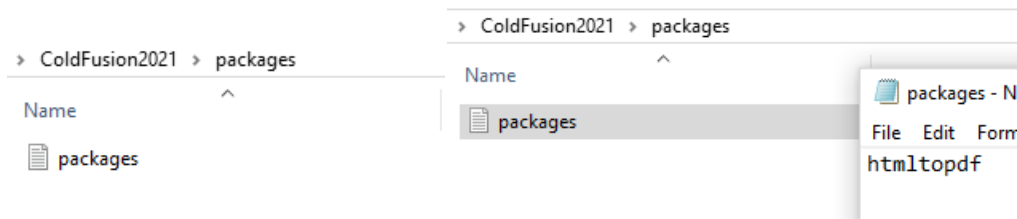
The first one is the list command. This will display every installed package. Simply type “list” at the prompt. The current packages installed will return as green text.

```
cfpm>list
The packages repository https://cfmodules-dev.stage.adobe.com/cf_main/bundlesdependency.json is not accessible
d the packages that are available locally in the C:\ColdFusion2021\bundles directory.
adminapi, version : 2021.0.0.322509
administrator, version : 2021.0.0.322509
cfpm>
```

Option 2 (Listall Command): One other very useful option is the “listall” command. This will provide you with a list of every package in the download repository. The items in yellow are ones that are available for download.

```
cfpm>listall
The packages repository https://cfmodules-dev.s
d the packages that are available locally in th
adminapi, versions : [2021.0.0.322509]
administrator, versions : [2021.0.0.322509]
ajax, versions : [2021.0.0.322509]
awsdynamodb, versions : [2021.0.0.322509]
awslambda, versions : [2021.0.0.322509]
awss3, versions : [2021.0.0.322509]
awss3legacy, versions : [2021.0.0.322509]
awssns, versions : [2021.0.0.322509]
awssqs, versions : [2021.0.0.322509]
axis, versions : [2021.0.0.322509]
azureblob, versions : [2021.0.0.322509]
azureservicebus, versions : [2021.0.0.322509]
caching, versions : [2021.0.0.322509]
cfmongodb, versions : [2021.0.0.322509]
chart, versions : [2021.0.0.322509]
com, versions : [2021.0.0.322509]
db2, versions : [2021.0.0.322509]
debugger, versions : [2021.0.0.322509]
```

Option 3 (Installing Packages via a File): This could be tremendously useful when automating repeatable steps in a pipeline. The file could be part of the build process and committed to version control (or Artifactory) and executed by your CI server. For this example, create a file with the list of packages and execute the command. At the command prompt enter “install [location]\packages.txt”.



Option 4 (Scanandinstall Packages): As noted earlier, ColdFusion is now modularized with the ability to install standalone packages as needed for your application. As one could imagine, the task to identify which packages are essential could be very difficult, especially if your application is dependent on many packages. Without the expertise of what is vital, a game of whack-a-mole could ensue.

ColdFusion 2021 now ships with a code scanner. This will automatically scan your specified code base and will return recommended packages. By simply providing the path to your code, site url and port, ColdFusion Code Scanner will ensure your product builds and ships with exactly what it needs.

This certainly simplifies things and one more great example of how ColdFusion 2021 can integrate seamlessly into your pipeline. This feature can be completely automated in your CI Build process, and repeatable on every deployment. When new code is deployed, you can rely on the code scanner to ensure the proper packages are installed and running, without the headache of figuring out what is needed to keep things light and fast.

- a) Enter the following command: `scanandinstall <path_to_your_code_base> http://[CF_SERVER_HOST]:port`
- b) For example, `scanandinstall <code_base/wwwroot/cfm> http://localhost:8500`

Option 5 (Non-interactive Install): If you are looking for true automation of the package manager, which should be the case if you are building for a pipeline process, the CFPM utility non-interactive mode can be used to accomplish the build. Open your command prompt and enter: `cfpm COMMAND ARGUMENTS` For example: `C:\>cfpm install <package_name>`

In review, several new features in ColdFusion 2021 have been covered with respect to installation procedures. There are many configuration options you can select. Many of the interactive steps that were once required for a ColdFusion installation are now available on the command line.






For sake of automation in your pipeline, these enhancements are great candidates to be introduced into your CI/CD process. You will no longer be required to manually configure and set up servers during your build and deployment process. Combining these benefits with a solid version control system, testing, and a build process will get you closer to your ultimate CI/CD goal.

CFSetup

ColdFusion 2021 offers a CFSetup command line utility to manage all your ColdFusion settings. Configuring your ColdFusion Administrator settings, making your configurations scriptable, and exporting and importing settings from different instances in your organization can be accomplished through this method.

The CFSetup tool is in the following directory:

`/coldfusion/config/cfsetup`

> ColdFusion2021 > config > cfsetup				
Name	^	Date modified	Type	Size
 cfsetup		10/26/2020 8:33 PM	Windows Batch File	2 KB
 cfsetup		10/26/2020 8:33 PM	Executable Jar File	803 KB
 findjava		10/26/2020 8:33 PM	Windows Batch File	1 KB
 log4j.properties		10/26/2020 8:33 PM	PROPERTIES File	1 KB
 proposedSettings		10/26/2020 8:33 PM	JSON File	34 KB

You have three different options when setting up your ColdFusion service. In the tutorial below we will highlight the stand-alone mode.

Before we do that, it is important to cover the script mode. Since a pipeline script will most likely be executed via a build process, manual intervention should not be a requirement as it will defeat the purpose.

The ability to define a sequence of commands in a batch file allow you to automate your pipeline build process, configuring all settings in a repeatable structure.

The options are vast, but a simple example should suffice in getting the initial foundational pieces in place.

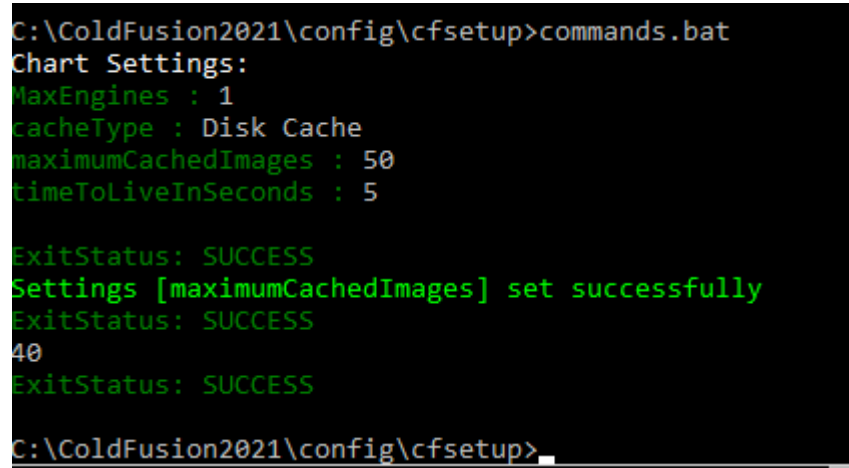
Create a commands.bat file and enter following commands and save the file.

```
@echo off
```

```
call cfsetup.bat show chart C:\ColdFusion2021\cfusion
```

```
call cfsetup.bat set chart maximumCachedImages=40 C:\ColdFusion2021\cfusion
```

call cfsetup.bat get chart maximumCachedImages C:\ColdFusion2021\cfusion



```
C:\ColdFusion2021\config\cfsetup>commands.bat
Chart Settings:
MaxEngines : 1
cacheType : Disk Cache
maximumCachedImages : 50
timeToLiveInSeconds : 5

ExitStatus: SUCCESS
Settings [maximumCachedImages] set successfully
ExitStatus: SUCCESS
40
ExitStatus: SUCCESS
C:\ColdFusion2021\config\cfsetup>
```

One more example to reiterate the flexibility and power of scripting the build process. We can change the administration password through the same .bat file.

Open the same commands.bat file and enter the following:

set security admin=newpassword cfusion

Be sure to restart your server after setting a configuration.

There are many configurations and options that can be configured. The “Help” option on the command line can assist as well.

```

C:\ColdFusion2021\config\cfsetup>cfsetup.bat help
Usage: [COMMAND]
Description: ColdFusion setup tool.
Commands:
  help      Command to get help for any category. List commands `help`, list categories `show
             category`
             Usage: help [<category>]<command>]
  alias     Command to add a ColdFusion instance as alias and update/delete that alias.
             Usage: alias <alias name> [<cfhome>]
  select    Command to select an aliased ColdFusion instance.
             Usage: select <aliasname>
  deselect  Command to deselect an aliased ColdFusion instance.
             Usage: deselect
  get       Command to get value of a setting of any category for an instance.
             Usage: get <category> [<servicename>] <settingKey> [<aliasname|cfusionhome>]
  set       Command to set a setting of any category for a provided/selected instance.
             Usage: set <category> [<servicename>] <settingKey=settingVal> [<cfusionHome|aliasname>]
  show      Command to show all settings of a category from a provided/selected ColdFusion instance

```








Adobe ColdFusion Docker Image

Although the steps below will highlight the ColdFusion installer, this would also be a great opportunity to exercise the ColdFusion Docker image published by Adobe. The ability to spin up a development environment, and have the same consistent build for other developers, and possibly staging or production servers, simplifies the process. It should be investigated and determined if it makes sense for your organization.

Creating REST Web Services with ColdFusion

You can create and publish a ColdFusion component (CFC) or any function in a component as a REST Resource.

1. First go to the root directory of your ColdFusion installation and create a directory for your service calls. The ColdFusion Components (CFCs) will live in this directory. For this example, you can create a directory called webservices.

) > ColdFusion2021 > cfusion > wwwroot >				
Name	^	Date modified	Type	Size
 cf_scripts		10/26/2020 8:57 PM	File folder	
 CFIDE		10/26/2020 8:57 PM	File folder	
 restplay		10/26/2020 8:57 PM	File folder	
 WEB-INF		10/26/2020 9:13 PM	File folder	
 webservices		2/2/2021 9:12 PM	File folder	
 crossdomain		10/26/2020 8:32 PM	XML Document	1 KB
 index.cfm		11/7/2020 9:08 PM	CFM File	1 KB

2. When creating a Component (CFC) specify the cfcomponent tag to one of the following: restPath or rest.

In this example we will use restPath and set it equal to "customers".

We have set the rest attribute equal to "true".

And for good practice we have set the name attribute equal to "Customers".

Tag Based Syntax:

```
<cfcomponent rest="true" restpath="customers" Name="Customers">
```

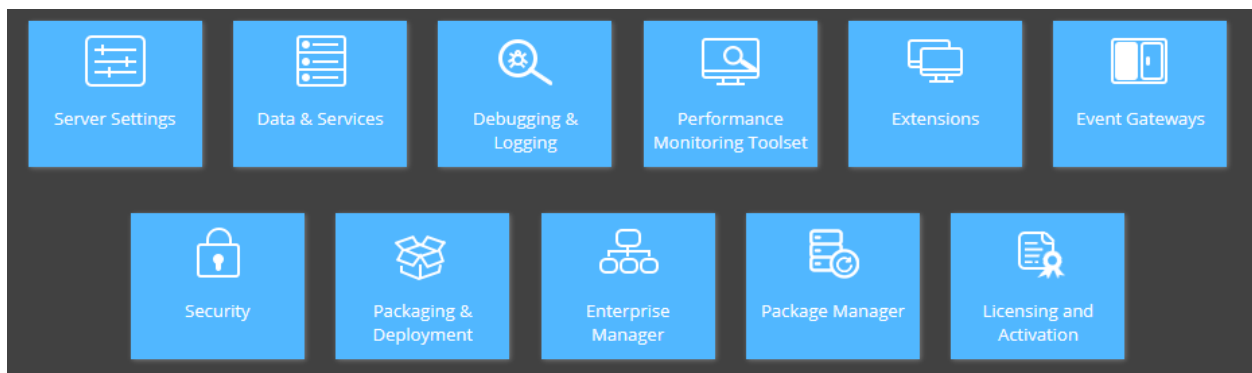
Script Based Syntax:

```
component rest="true" restpath="/customers" Name="Customers">{
```

Next, you must register your REST API Services in ColdFusion. We will cover two options on how to complete this. We will assume you have access to the ColdFusion Administrator and cover that scenario first. The second option can be done directly through code in ColdFusion, in the event you do not have access to the administrator.

ColdFusion Administrator (REST Service Registration)

1. Log into the ColdFusion Administrator and click on the Data & Services. You should be able to access the administrator by visiting this link (<http://localhost:8500/CFIDE/administrator/index.cfm>).



2. Click on the REST Services option

Data & Services

Data Sources NoSQL Data Sources ColdFusion Collections Solr Server Web Services REST Services

3. Click on the Browse Server option and select the directory where your Rest Components will live. Earlier, we created the webservices directory in our ColdFusion root installation. Let's continue to use this for demo purposes.

Register your applications and folders. ColdFusion automatically registers CFCs found in the registered folders.

Add/Edit REST Service

Root path

Browse Server

Application path or root folder where CFCs reside

Host [:Port]

Host name for the REST service(localhost is default). Example: localhost:8500 (Optional)

Service Mapping

Alternate string to be used for application name while calling REST service. (Optional)

Example: http://{Hostname}:{Port}/{REST Path}/{Service Mapping}/{Component REST Path}

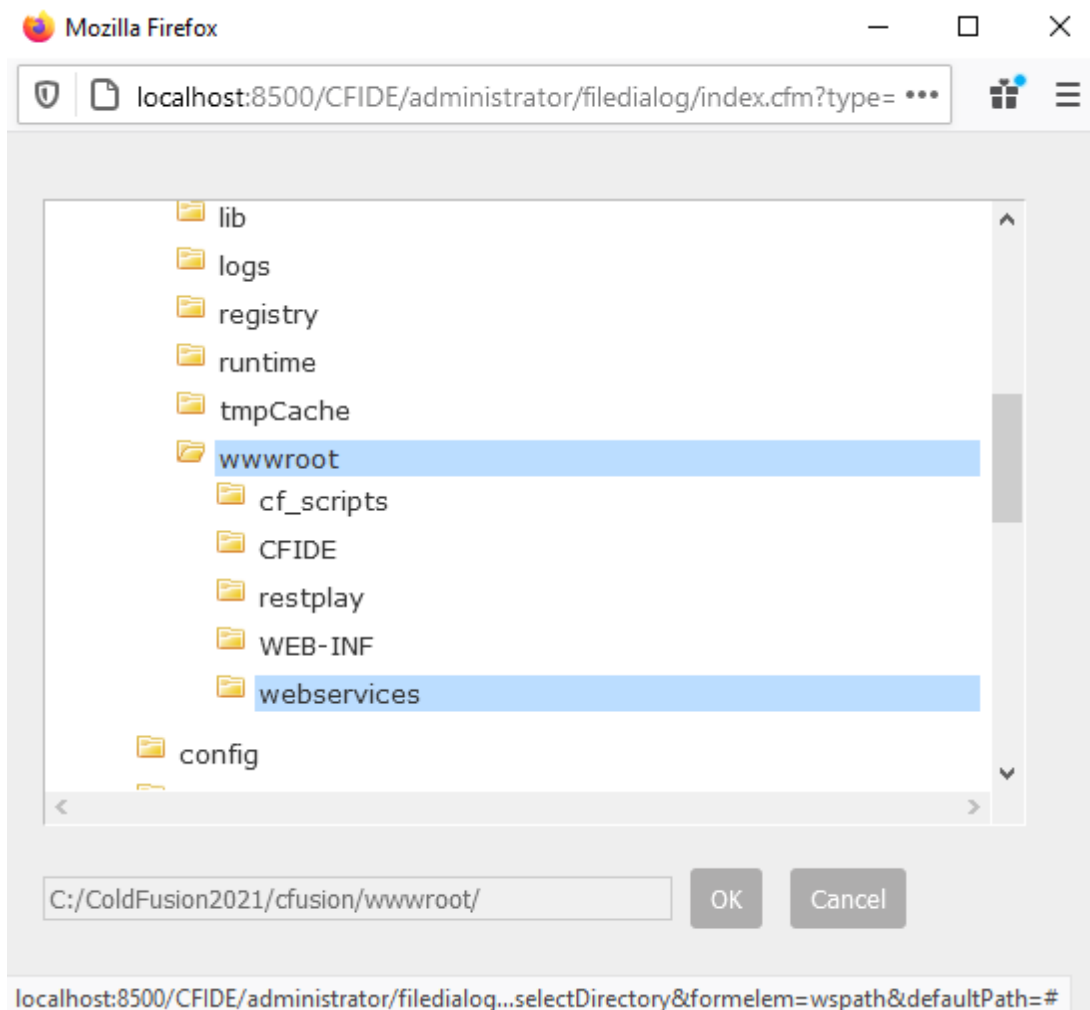
Set as default application

☐

Set an application as default to exclude the application name in the URL while calling the web service. One default application is allowed for a host.
Example http://{Hostname}:{Port}/{REST Path}/{Component REST Path}

Add Service

4. Select the directory webservices and click on OK.



5. Select the service mapping name. In this tutorial we will call it "api", and we will set it as the "default application" and click on "Add Service"

Register your applications and folders. ColdFusion automatically registers CFCs found in the registered folders.

Add/Edit REST Service

Root path

Application path or root folder where CFCs reside

Host [:Port]

Host name for the REST service(localhost is default). Example: localhost:8500 (Optional)

Service Mapping


Alternate string to be used for application name while calling REST service. (Optional)
Example: http://{Hostname}:{Port}/{REST Path}/{Service Mapping}/{Component REST Path}

Set as default application ☒

Set an application as default to exclude the application name in the URL while calling the web service. One default application is allowed for a host.
Example http://{Hostname}:{Port}/{REST Path}/{Component REST Path}

6. After the service is created, you should see the meta data associated with it listed in the "Active ColdFusion REST Services". This completes the Administration configuration.

Active ColdFusion REST Services

Actions	Root Path	Service Mapping	Default	Host:Port
  	C:\ColdFusion2021\cfusion\wwwroot\webservices	api	YES	

Update REST Path

Change this settings to get customized URL. For example, if you change this setting to 'comservices', URL would look like http://{Hostname}:{Port}/comservices/{ServiceMapping}/{Resource REST Path}










7. The contextual endpoint for your services can now be invoked by calling <http://localhost:8500/rest/api/{Mapping and Sub Resources}>

We will now cover one additional option if your access to the ColdFusion administrator is not available.

ColdFusion Programmatic REST API Registration (Application.cfc)

1. First you will need to add a Application.cfc to the root directory of your installation.

> ColdFusion2021 > cfusion > wwwroot

Name	^	Date modified	Type	Size
 cf_scripts		10/26/2020 8:57 PM	File folder	
 CFIDE		10/26/2020 8:57 PM	File folder	
 coderegister		2/7/2021 6:04 PM	File folder	
 restplay		10/26/2020 8:57 PM	File folder	
 WEB-INF		2/2/2021 9:50 PM	File folder	
 webservice		2/6/2021 10:23 PM	File folder	
 Application		2/7/2021 6:05 PM	CFC File	1 KB
 crossdomain		10/26/2020 8:32 PM	XML Document	1 KB
 index		11/7/2020 9:08 PM	CFM File	1 KB

2. Add the "restInitApplication" function and provide the path to your directory where your service components are located.

Utilizing the `getDirectoryFromPath` function and `getCurrentTemplatePath` function, you can dynamically set these values and then simply specify the "Service Mapping" name. For demonstration purposes I have named this sample "coderegister" so it will not cause conflict with the previous Administrator registration.

In the earlier part of the tutorial we utilized the Administrator directly. In this case, the Application.cfc will register the services in the system, and they will appear in the Administrator automatically.

```
component
{
    this.name = "Sample REST API's";
    this.applicationTimeout = CreateTimeSpan( 5, 0, 0, 0 );
    this.sessionManagement = true;
    this.sessionTimeout = CreateTimeSpan( 0, 0, 45, 0 );
    this.restsettings.cfclocation = ".*.*";
    this.restsettings.skipcfcwitherror = true;

    public boolean function onApplicationStart()
    {
        restInitApplication(getDirectoryFromPath(getCurrentTemplatePath()), "coderegister");
        return true;
    }
}
```

If you do have access to the ColdFusion Administrator, you can verify that the REST service has been registered and is active. You can see below the “coderegister” service group is now available and ready to take requests.

Active ColdFusion REST Services



Actions	Root Path	Service Mapping	Default	Host:Port
  	C:\ColdFusion2021\cfusion\wwwroot\webservices	api	NO	
  	C:\ColdFusion2021\cfusion\wwwroot\coderegister	coderegister	NO	

3. The contextual endpoint for your services can now be invoked by calling <http://localhost:8500/rest/coderegister/{Mappings and Sub Resources}>

Sample Components and Functions to build your REST APIs

To illustrate some sample code, we will continue with the samples we have already provided. In the webservices directory that we mapped in ColdFusion earlier, we will create a Component (CFC) called Customers.

We will also create a CustomersScriptBased Component (CFC) to illustrate the same process but without tags. If you prefer to write only in script those samples will be provided as well.

) > ColdFusion2021 > cfusion > wwwroot > webservices				
Name	^	Date modified	Type	Size
 Customers		2/6/2021 10:09 PM	CFC File	2 KB
 CustomersScriptExample		2/6/2021 11:05 PM	CFC File	2 KB

Let's take a closer look at the Customers.cfc. We will begin by writing a simple function called get, and in that an array to store all customers. Take note of some key attributes that we have specified in the CFC.

For the Customers.cfc we have used the syntax below (tag based):

```
<cfcomponent rest="true" restpath="customers" Name="Customers">
```

The rest argument is equal to true and the restpath was set to customers. It is not required to match the name of the CFC. In this case, we will keep it simple and consistent.

Next, we look at the function tag. It is here we declare the name of our method and call it "get". We set the access equal to "remote". Even though this will return a JSON payload the data type is set to "array", since ColdFusion handles the serialization of JSON automatically. We make sure to set the produces equal to "application/json" and lastly the httpmethod is set to "GET".

One additional thing to understand, we did not set the restPath for the get. Since you will invoke this from the URL, and there is no other httpMethod of GET in the entire class, the routing will not be an issue.

ColdFusion will know to invoke this class and method and process the request. If we did have multiple GETs in the CFC this would be a problem. For good practice, it makes logical sense to provide the restPath at a function level.

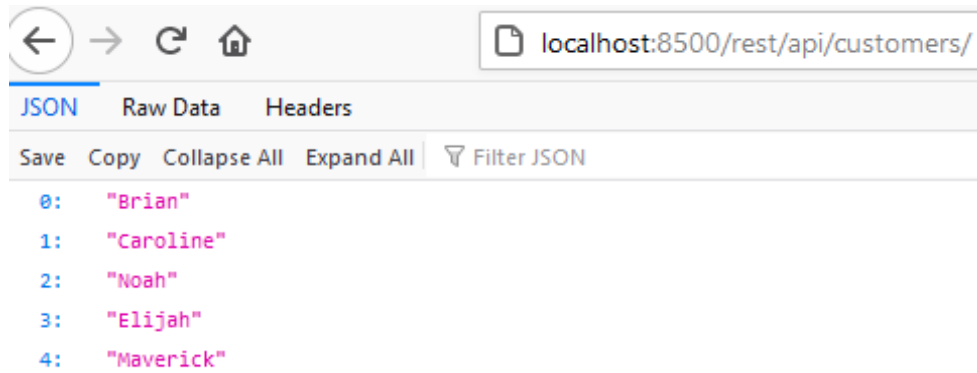
GET API Call Tag Based

```
//GET Request to return customers
<cffunction name="get" access="remote" returnType="array" produces="application/json" httpmethod="GET">
    <cfset customers = arrayNew(1)>
    <cfset customers[1] = "Brian">
    <cfset customers[2] = "Caroline">
    <cfset customers[3] = "Noah">
    <cfset customers[4] = "Elijah">
    <cfset customers[5] = "Maverick">
    <cfreturn customers />
</cffunction>
```

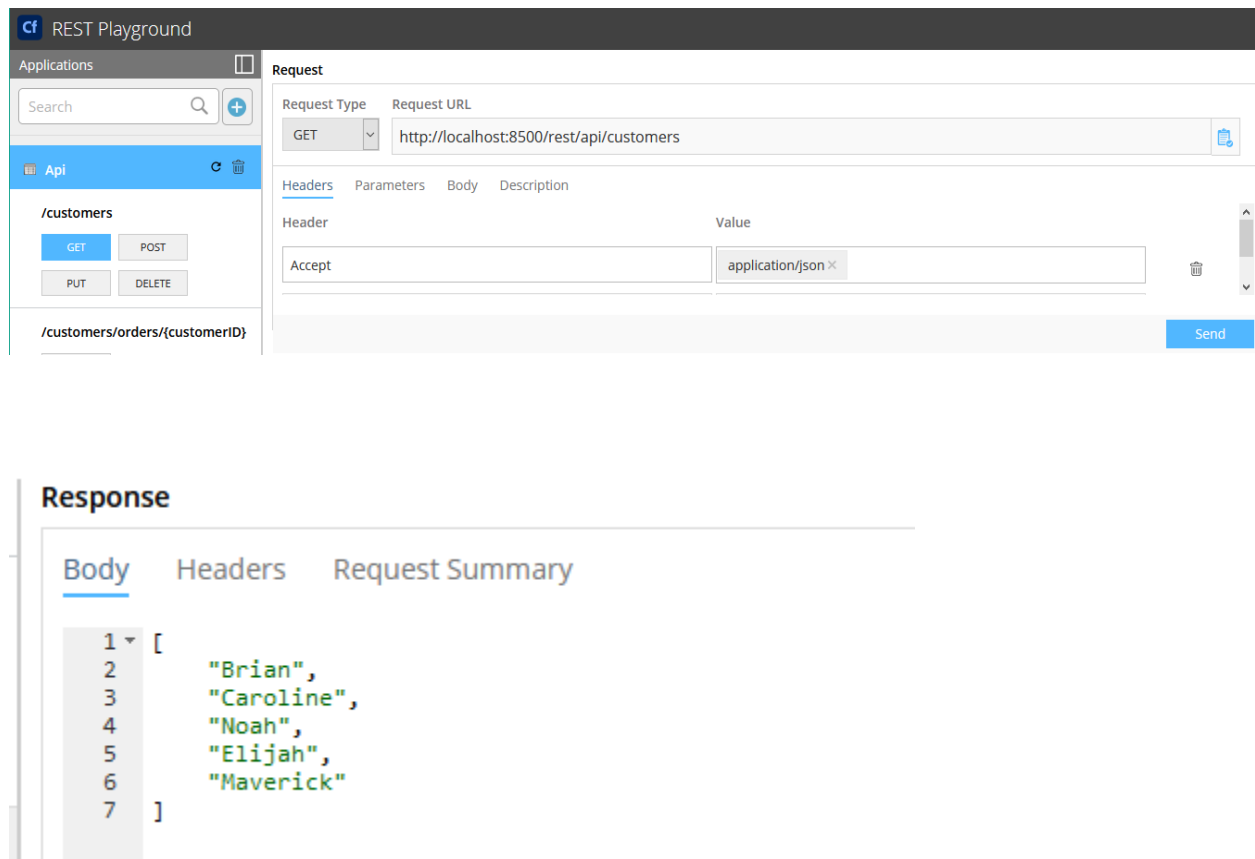
GET API Call Script Based

```
//GET Request to return customers
function get () access="remote" returnType="array" produces="application/json" httpmethod="GET" {
    var customers = [];
    customers[1] = "Brian";
    customers[2] = "Caroline";
    customers[3] = "Noah";
    customers[4] = "Elijah";
    customers[5] = "Maverick";
    return customers;
}
```

If you invoke from the browser you can see the data is returned in a JSON format.



Invoking it from the Adobe ColdFusion REST Playground will return the same response, with a few more details.



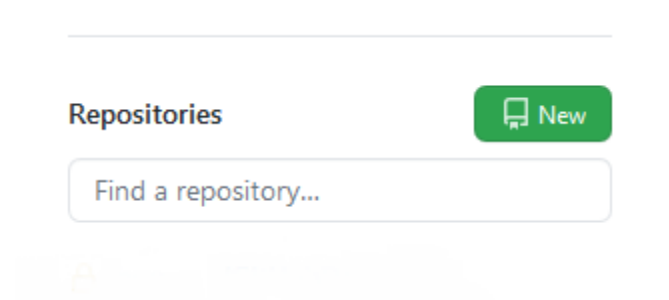
We will be using this REST API later in the tutorial as a health check for your Jenkins pipeline.

Source Control

When it comes to source control, there are a few solid options to choose from. Git, Concurrent Versions System (CVS), and Apache Subversion (SVN), are a few of the more dominant ones in this space. In the sample below, Git was the selected tool for version control.

Step 1: Create a new repository

This tutorial will assume you already have a Git account. Once you login create a New Repository.





Step 2: Fill in the details of your new repository

Provide a name and settings based on your configuration needs.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)



Owner ^{*} Repository name ^{*}

 bsappey ▾ / coldfusion-stratus 

Great repository names are short and memorable. Need inspiration? How about [scaling-guide?](#)

Description (optional)

ColdFusion Circle CI Pipeline Example

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

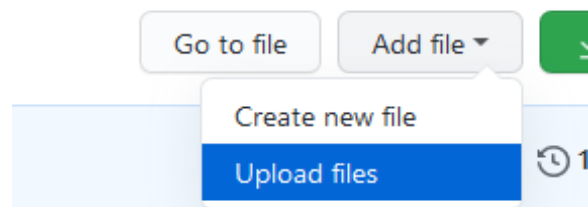
Initialize this repository with:

Skip this step if you're importing an existing repository.

- ☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)
- ☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)
- ☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

Step 3: Successful repository created


For this example, we will use the “Add File” menu and move our local files to Git. There are number of options when adding objects to your repository. In this case, we will manually upload the files to Git. In best practice, you would be implementing some version of a Git workflow. There are quite a few options, and of course, you can always create a customized workflow that meets your needs.



Step 4:

crossdomain.xml

index.cfm



Commit changes

Add files via upload

Add an optional extended description...

☒ Commit directly to the `main` branch.

☐ Create a **new branch** for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel

Step 5: Successful commit of files from local

main


1 branch

0 tags

Go to file

Add file

Code

 bsappey

Add files via upload

55c1776 now

2 commits

README.md	Initial commit	3 minutes ago
crossdomain.xml	Add files via upload	now
index.cfm	Add files via upload	now

README.md

coldfusion-stratus

ColdFusion CI Tutorial

Continuous Integration Server

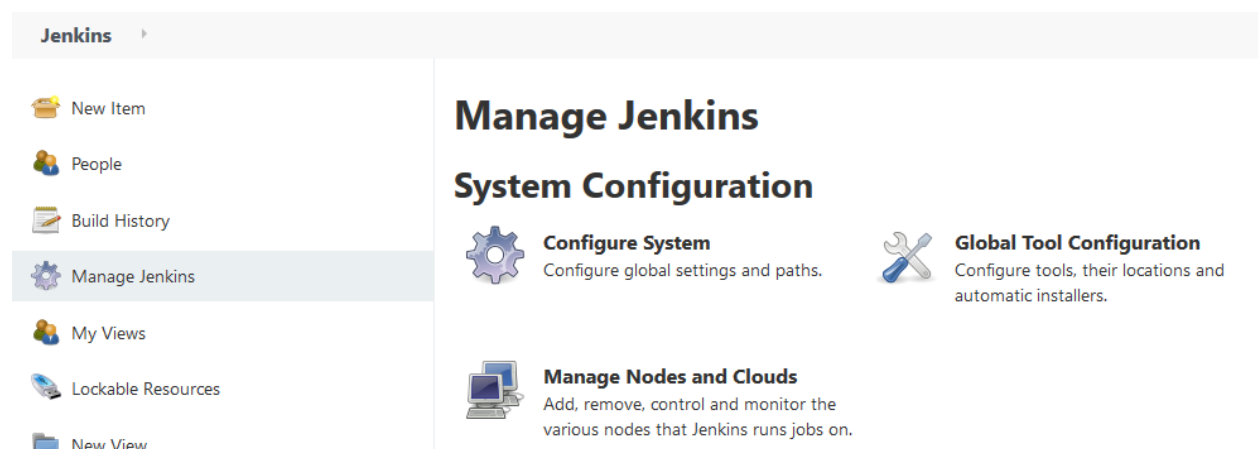
One of the most crucial parts of the CI process is selecting the tool that is right for you. Jenkins will be highlighted in this tutorial but it is important to note, a lack of tools is not an excuse. CI can be performed without any of the CI tools on the market. Jenkins, Travis CI, Circle CI, and GitLab CI, are just a few of the options available that can be explored for building out your CI server.

To keep things simple, this tutorial will run a local copy of Jenkins. There are many configurations and options when running Jenkins. You could write your own build scripts, as well as download plugins from the Jenkins Plugin Manager.

After downloading Jenkins and installing it locally, you are ready to construct your first build.

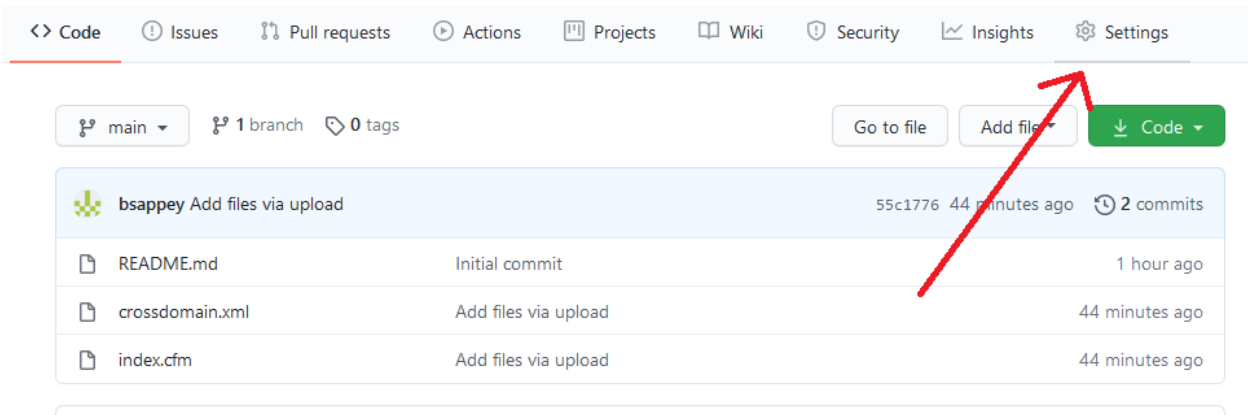
Step 1: Manage Jenkins Configurations

After logging into Jenkins, you can customize the settings of the application and the build process. There are several options that can be configured, including plugins. In this tutorial, Git commits will be automatically detected in Jenkins.



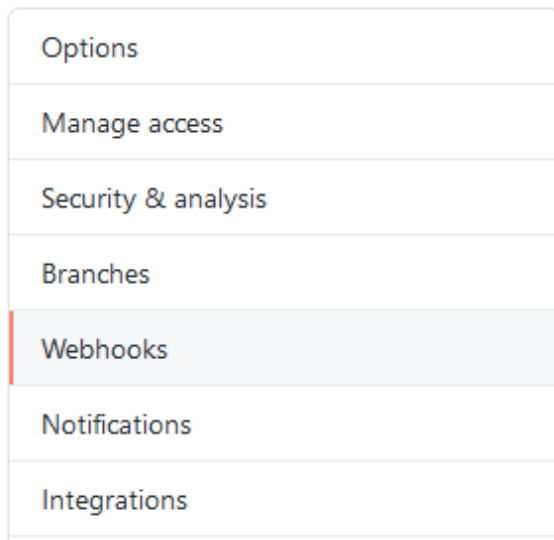
Step 2: Git Settings and Webhooks

In order to create a connection with Jenkins, we need to configure Git to communicate any change. Log into your Git repository that you created earlier and click on the Settings option on the top right menu.



Step 3: Git Webhooks Configuration

Click on the Webhooks left navigation.



Webhooks


Webhooks allow external services to be notified of the URLs you provide. Learn more in our [We](#)

In the Payload URL, specify your Jenkins Server URL. Be sure to include the `/github-webhook/` as part of the string. Select the Content Type as needed. The webhook will fire when an event is triggered. You can configure which events will trigger a call to Jenkins.

Security & analysis	<div>Payload URL *</div> <div>http://localhost:8080/github-webhook/</div> <div>Content type</div> <div>application/x-www-form-urlencoded ↕</div> <div>Secret</div> <div></div> <div>Which events would you like to trigger this webhook?</div> <div><input type="radio"/> Just the push event.</div> <div><input type="radio"/> Send me everything.</div> <div><input checked="" type="radio"/> Let me select individual events.</div>
---------------------	--


Step 4: Jenkins and Creating a New Item


Switching back over to Jenkins, you can now start to create your first build. Use the select “New Item” option on the left menu.




Jenkins


Jenkins ▶


 New Item

 People

New Item

 Build History

 Manage Jenkins

 My Views

Welcome to Jenkins!

Create an agent or configure a cloud to set up

Create a job to start building your software project.

Step 5: Select a pipeline and provide a name

In this example, select "Pipeline" and provide a name.



Enter an item name

ColdFusion2021CI

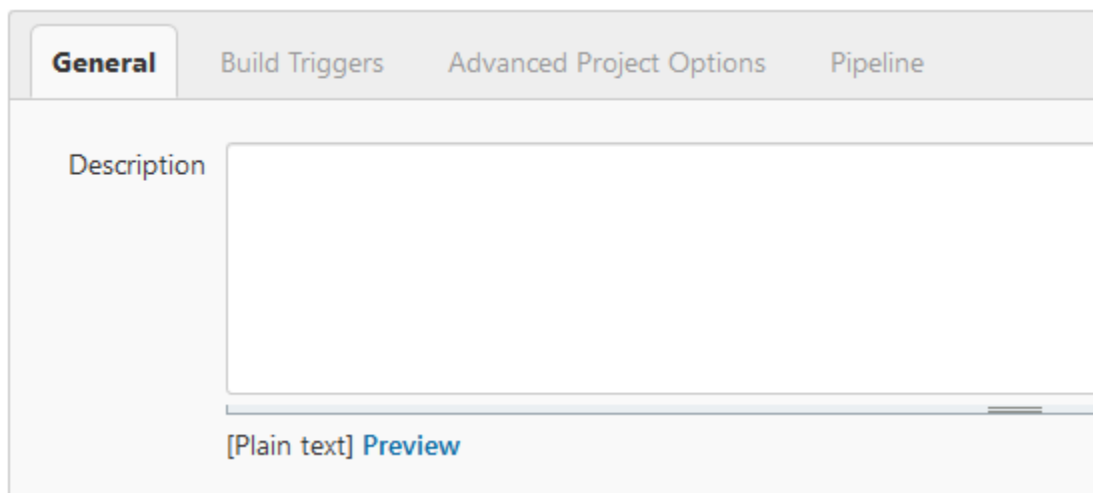
» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Step 6: Configure the pipeline using script

After you create the item, you can begin to configure the Pipeline through script. Select the "Pipeline" option on the sub navigation.



General Build Triggers Advanced Project Options Pipeline

Description

[Plain text] [Preview](#)

Step 7: Creating a basic pipeline script

Click on the "Pipeline" menu item and create a basic pipeline script. In the sample below, the script will perform a simple checkout if a commit is detected in the Git Repository. The "stage" name can be any string to identify which Step/Stage has been executed and the status.

Advanced Project Options

Advanced...

Pipeline


Definition Pipeline script

Script

```
1 node{
2   stage('SCM Checkout'){
3     checkout([$class: 'GitSCM',
4               branches: [[name: '**']],
5               doGenerateSubmoduleConfigurations: false,
6               extensions: [],
7               submoduleCfg: [],
8               userRemoteConfigs: [[credentialsId: ,
9                                   url: 'https://github.com/bsappey/coldfusion-stratus']]])
10  }
11 }
```


Step 8: Initiate the Build Process


Click on the Item Name you created to return to the Pipeline Screen.





Jenkins

Jenkins > ColdFusion2021 CI


 Back to Dashboard

 Status

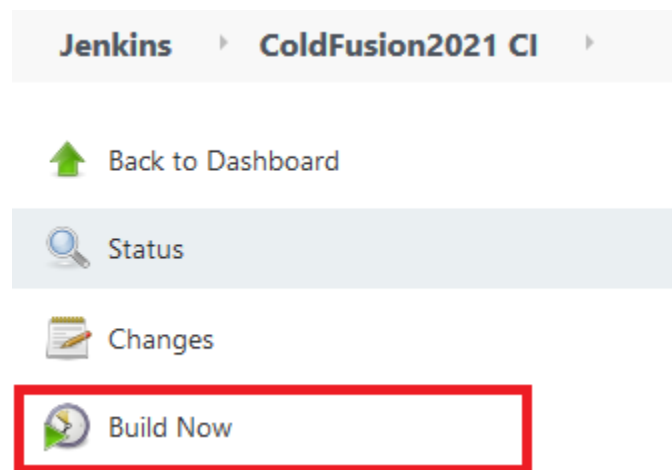
 Changes

 Build Now

Pipeline ColdFusion2021 CI

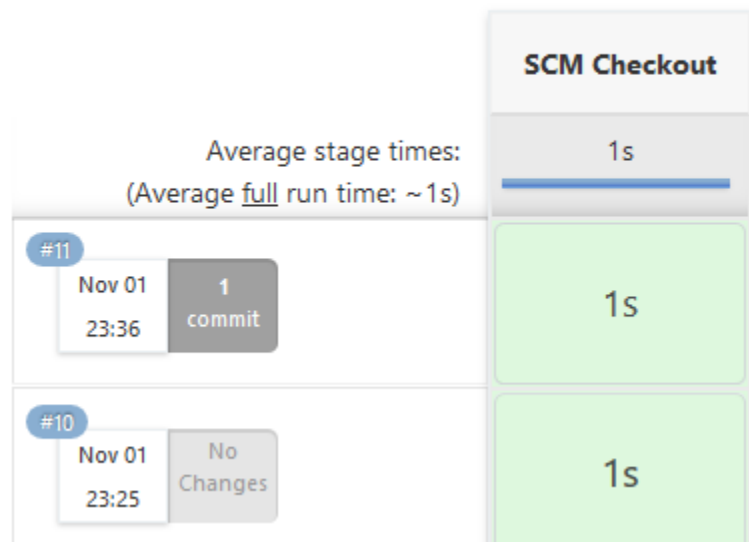
 Recent Changes

Step 9: Click on the “Build Now” button to start the build process.



Step 10: The build will initiate, once it completes a status of success or failure will render under the “Stage” name you created in script. In this example, Jenkins will perform a checkout on any commits to the repository that we configured in Git.






Stage View



Step 11: Confirm or access the Git checkout

A successful Checkout via the pipeline script will now drop a copy of the code into your Jenkins workspace. Navigate to the directory and verify the files that were transferred.

) > Windows > System32 > config > systemprofile > AppData > Local > Jenkins > .jenkins > workspace > ColdFusion2021 CI

Name	Date modified	Type	Size
 crossdomain	10/29/2020 11:57	XML Document	1 KB
 index.cfm	10/29/2020 11:57	CFM File	1 KB
 README.md	10/29/2020 11:57	MD File	1 KB
 test.cfm	11/1/2020 10:40 PM	CFM File	1 KB
 test2.cfm	11/1/2020 11:36 PM	CFM File	1 KB

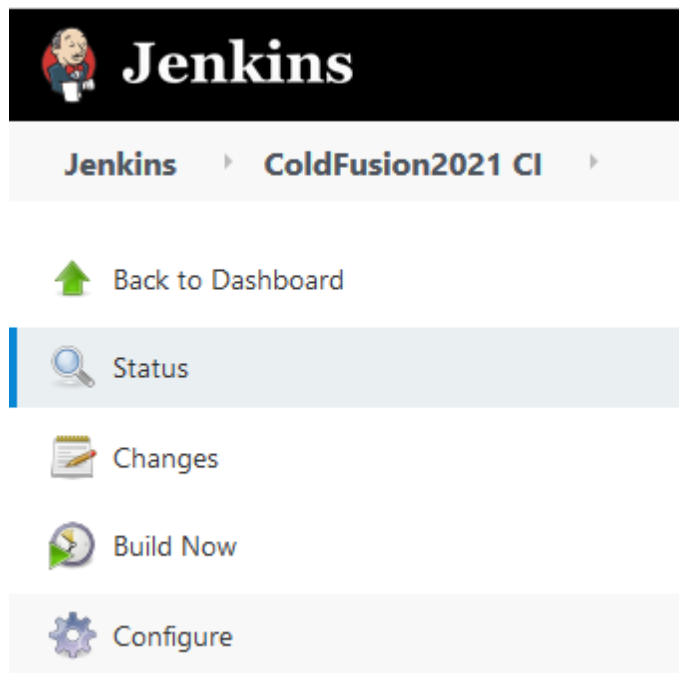
Step 12: Workflows, Builds, and Git Commits for the integration

Every workflow will vary, in most cases a developer would commit their local changes to a Feature branch. A Git checkout would most likely occur against a Master branch. For this tutorial, we will assume the code made it to the Master branch. It should be noted, that with the webhooks enabled, you would not have to hit the "Build Now" option. As soon as a commit is detected, the Build process will execute automatically.

With the assumptions made, the next addition to the pipeline will be integrating the code.

Step 13: Configure and Create the SSH Script

On the left menu select "Configure".



Step 14: SSH Pipeline Script

Select the "Pipeline" sub nav option and write your script.

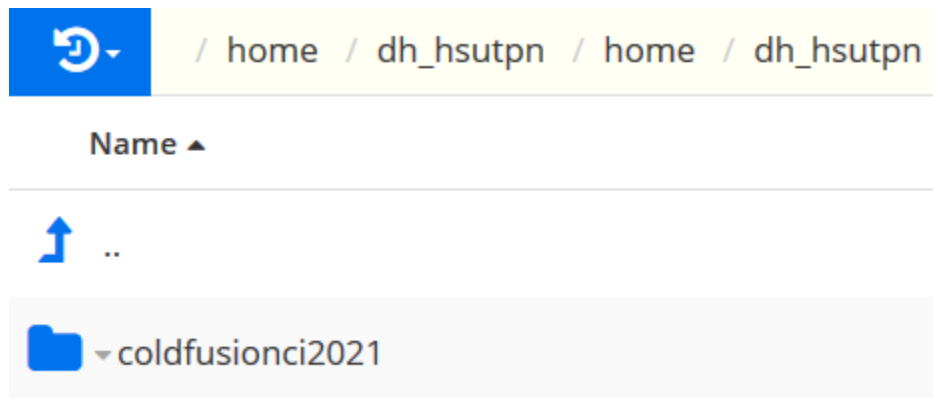
General	Build Triggers	Advanced Project Options	Pipeline
Pipeline			
Definition		Pipeline script	

Below is a sample SSH command. In this example you will need to provide your Remote Directory and SSH credentials. Provide a string name for the stage, this again will allow for visibility and reporting when the builds execute. This script will grab the files in your Jenkins Workspace and deploy them to your integrated development environment.

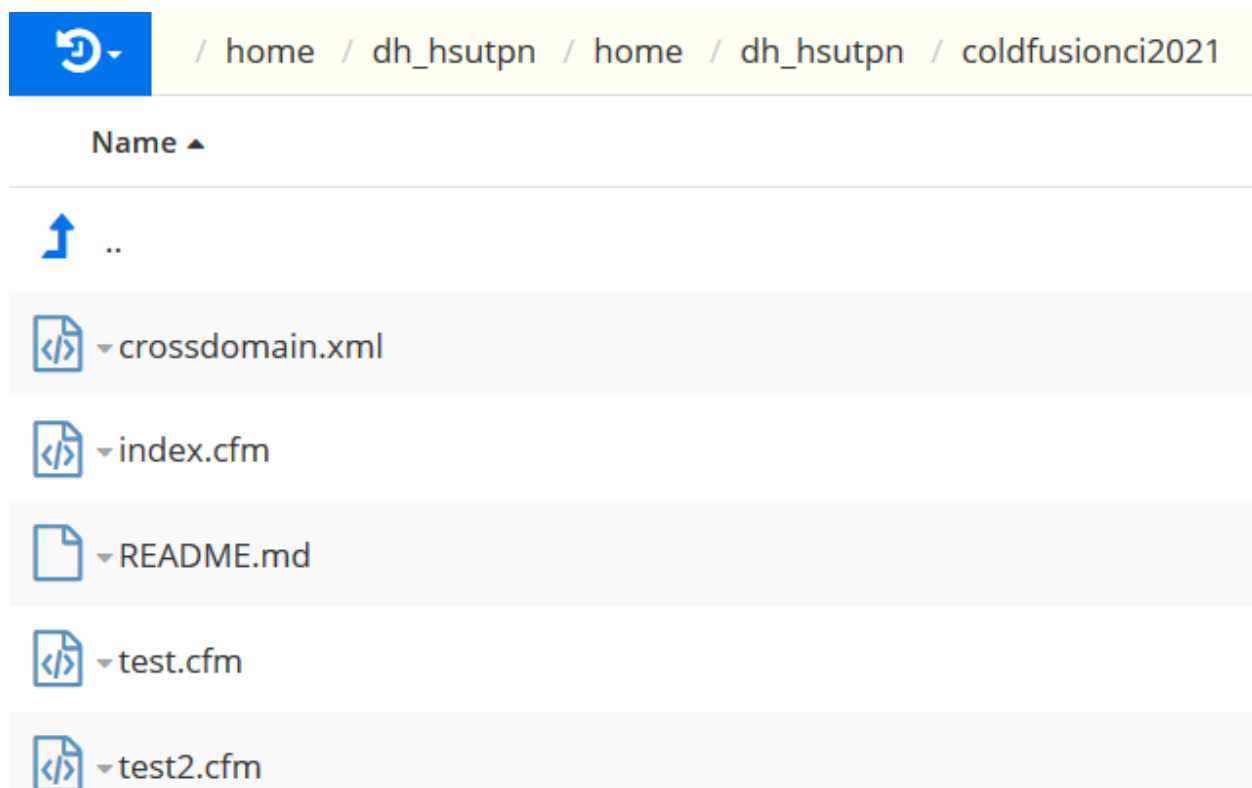
```

10     }
11     stage('SSH File'){
12         sshPublisher(publishers: [sshPublisherDesc(configName: 'Dream',
13         transfers: [sshTransfer(cleanRemote: false,
14         excludes: '',
15         execCommand: '*',
16         execTimeout: 120000,
17         flatten: false,
18         makeEmptyDirs: true,
19         noDefaultExcludes: false,
20         patternSeparator: '[, ]+',
21         remoteDirectory: '/home/dh_hsutpn/coldfusionci2021/',
22         remoteDirectorySDF: false, removePrefix: '', sourceFiles: '*')],
23         usePromotionTimestamp: false,
24         useWorkspaceInPromotion: false,
25         verbose: false)])
26     }
27 }
```

Verify that the files were successfully moved to your server

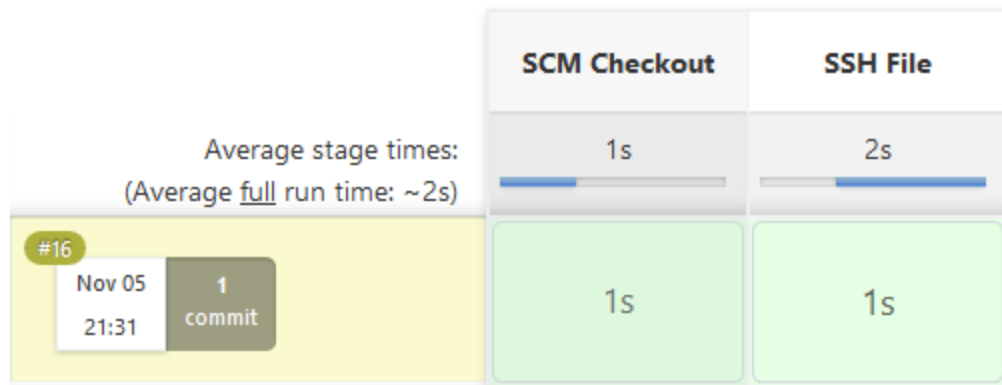


Opening the remote directory shows a successful integration of the files.



Lastly, you can verify that the build process ran successfully. As you can see the SCM Checkout (stage 1) completed, as well as the SSH File transfer (stage 2).

Stage View



ColdFusion 2021 and Development/Staging/Production Servers








This tutorial assumes that ColdFusion 2021 is already installed on your integrated development servers, staging/testing servers, and production servers. It is important to note, that a strong case should be made for containerization when building a pipeline. The Adobe Docker image for ColdFusion is certainly a great tool in helping with the pipeline.

With the assumption made that ColdFusion is already up and running, we can proceed with the next steps. It is possible to create a ColdFusion installation on a remote server using Jenkins if Docker is not in your current realm of tools.

Here is a sample script in the event you need to install and start ColdFusion. Utilizing the silent installer, you can run a command to install, unzip and start the ColdFusion Service. This stage could be placed anywhere in your build process.

Earlier we covered the properties file. If you did not change the file location or the name it can be found in the root of your download. The config file can be specified in the script for the silent installation.

ColdFusion2021

Name	Date modified	Type	S
 bundles	2/16/2021 9:55 PM	File folder	
 cfusion	2/17/2021 8:41 PM	File folder	
 config	10/26/2020 8:57 PM	File folder	
 jre	10/26/2020 8:57 PM	File folder	
 packages	10/26/2020 10:27 ...	File folder	
 Readme	10/26/2020 8:57 PM	File folder	
 config	10/26/2020 8:32 PM	Configuration sett...	

The installation script:

```
1 stage('CF2021 Silent Installer'){
2     bat 'C:\\ColdFusion2021\\cfusion\\bin --file-name config.properties --installer-mode silent'
3 }
4
```

Step 14: ColdFusion 2021 Package Manager

Now that you have successfully created a pipeline with two stages, let's look at some ColdFusion specific features that you can embed into your pipeline. After the files are integrated, you can scan the code and ensure all required packages are installed.

Select the "Pipeline" sub nav option and write your script.

GeneralBuild TriggersAdvanced Project OptionsPipeline

Pipeline

DefinitionPipeline script

To verify the commands, you can simply run the CFPD.bat file and "help" to display all options available. In the Jenkins console logs you can verify the results of a build. The stage name is below, configured to your specifications

```

25         verbose: false)))
26     }
27     stage('CFPM CLI Startup'){
28         bat 'C:\\ColdFusion2021\\cfusion\\bin\\cfpm.bat help'
29     }
30 }

```

As you can see, the build ran, and the .bat file successfully executed in the build process.

```

Stage Logs (CFPM CLI Startup)
Windows Batch Script -- C:\ColdFusion2021\cfusion\bin\cfpm.bat help (self time 1s)

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\ColdFusion2021 CI>C:\ColdFusion2021\cfusion\bin\cfpm.bat help

install PACKAGE[NAME[:VERSION]]
Installs a new package. If VERSION is not specified, then the latest version of the package will be installed. You can also specify comma-separated packages to install multiple packages.

install ALL
Installs all the available packages.

```

Since this pipeline build will be automated, you would not want to be concerned with having to run this manually.

One of the most powerful features of the CFPM utility is the "scanandinstall" option. It should be noted there are a lot of possibilities at this junction. You can certainly run these checks at a local development level as well. In fact, this process could be automated multiple times through the build.

You can keep it simple and execute a check immediately after the code is integrated into the development server. It's critical to make sure the file changes are not breaking, and the feedback loop is constantly updated and monitored.

```

26     }
27     stage('CFPM CLI Startup'){
28         bat 'C:\\ColdFusion2021\\cfusion\\bin\\cfpm.bat scanandinstall \\wwwroot\\ localhost:8500'
29     }
30 }

```

Step 15: Jenkins Review for SCM Checkout, SSH File Transfer, and CFPM Script Execution

Below you will find a condensed version of the three stages. You can see the three stages clearly broken out into ordered tasks.

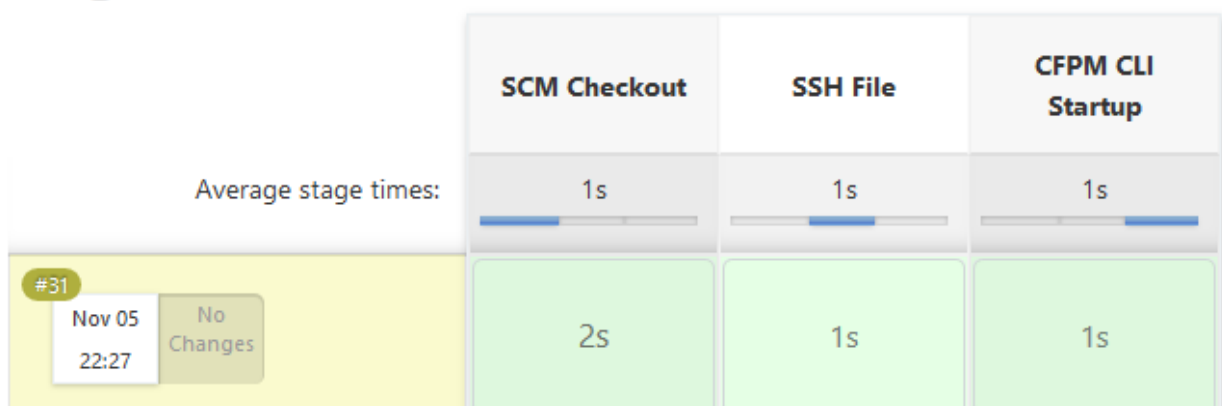
- The first stage will perform the Git checkout for any changes in the master branch.
- The second stage will SSH into your integrated dev server and copy the files, from the Jenkins and Git checkout.

- c) The third stage will run a quick check to ensure that all package dependencies have been installed.

```
node{
  stage('SCM Checkout'){
    checkout([$class: 'GitSCM',
      branches: [[name: '**']],
      doGenerateSubmoduleConfigurations: false,
      extensions: [],
      submoduleCfg: [],
      userRemoteConfigs: [[credentialsId: '[REDACTED]',
        url: 'https://github.com/bsappey/coldfusion-stratus']]])
  }
  stage('SSH File'){
    sshPublisher(publishers: [sshPublisherDesc(configName: 'Dream',
      transfers: [sshTransfer(cleanRemote: false,
        excludes: '',
        execCommand: '*',
        execTimeout: 120000,
        flatten: false,
        makeEmptyDirs: true,
        noDefaultExcludes: false,
        patternSeparator: '[, ]+',
        remoteDirectory: '/home/dh_hsutpn/coldfusionci2021/',
        remoteDirectorySDF: false, removePrefix: '', sourceFiles: '*')],
        usePromotionTimestamp: false,
        useWorkspaceInPromotion: false,
        verbose: false)])
  }
  stage('CFPM CLI Scan and Install'){
    bat 'C:\\ColdFusion2021\\cfusion\\bin\\cfpm.bat scanandinstall \\wwwroot\\ localhost:8500'
  }
}
```

Below is the Jenkins visual representation of the scripted stages above. As you can see all three ran successfully. Each one of the stages can be inspected for details about the stage.

Stage View



Step 16: Pulse Check After the Integration

One of the critical components of the CI process is the feedback loop. Immediately after you integrate the code, you need to ensure it is nonbreaking.

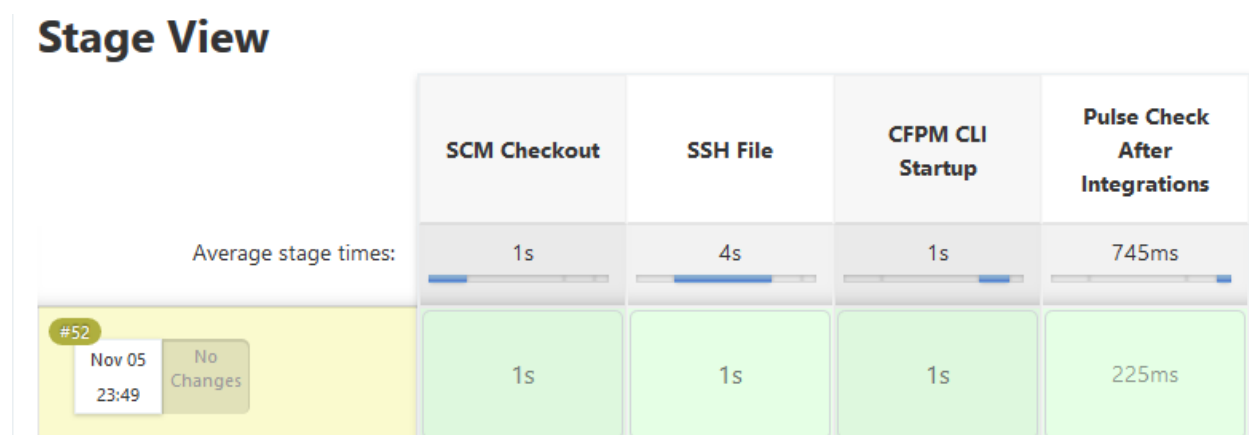
A simple pulse check will suffice in ensuring the environment is healthy and operating as expected. If an integration breaks, it is at this point that a developer would be able to assess and quickly make decisions on the next steps.

You could create a stage on a failed build that could revert the change or provide developers enough information to make a code change and apply a fix.

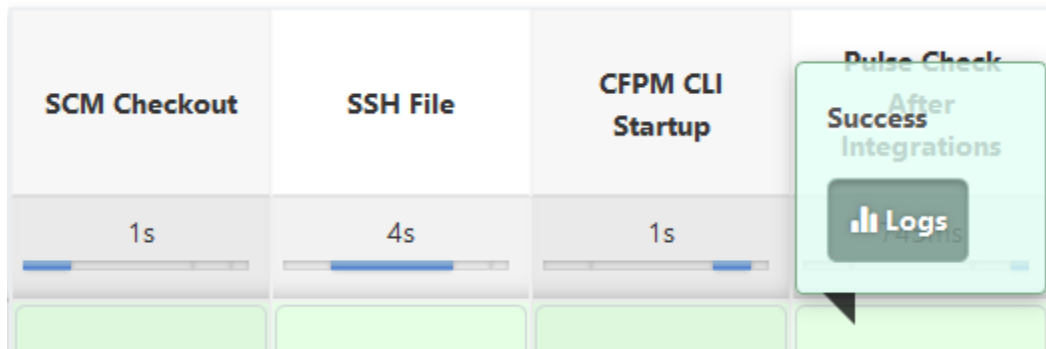
Below is a sample http request to verify if the code is returning a 200 for the http status code. If you remember earlier, we built a sample REST call. You could invoke a request and inspect the status.

```
36
37 ▾
38   stage('Pulse Check After Integrations'){
39     def response = httpRequest 'http://localhost:8500/rest/api/customers/'
40     if(response.status != 200){
41       System.exit(1)
42     }
43   }
```

Jenkins will visually show the stage “Pulse Check After Integrations” and provide feedback.



You can review the details of the build by hovering over any stage and clicking the Logs button in the tool tip.



Visually, you can review the http request and inspect it for any additional details you may need.

Stage Logs (Pulse Check After Integrations)

🔍 Perform an HTTP Request and return a response object -- `http://localhost:8500` (self time 76ms)

```
HttpMethod: GET
URL:
Sending request to url:
Response Code: HTTP/1.1 200 OK
Success code from [100..399]
```

Continuous Integration and Testing

With respect to testing, a pulse check is certainly not enough as a determining factor in whether you deploy the code to staging or production. There are many testing frameworks that can be leveraged, and this paper is not advocating the removal of this process from your pipeline.

Some of the testing frameworks that can be leveraged are Katalon, Selenium, and Appium (for mobile testing) to name just a few. Jenkins can integrate with the platforms to help your QA teams and engineers thoroughly test your code.

With API integrations or deployments, unit tests are an excellent way for developers to help test code through automation. These test cases (unit tests) can be deployed along with the code, and automatically triggered in the build plan. In some cases, this may be adequate to establish the health of the application.

Furthermore, load testing is an excellent item to incorporate as well. Each project may vary, but load testing new builds is a great way ensure successful deliveries. Load testing tools will provide extremely useful feedback for your applications performance during peak traffic times and during

extreme stress. Here are a few tools in the market today, LoadNinja, WebLoad, SmartMeter.io, and JMeter.

Performance Monitoring Toolset

As mentioned earlier ColdFusion 2018 released a Performance Monitoring Toolset (PMT). This might not be the most conventional use case, but there is value in exploring the PMT to help monitor your pipeline. When it comes to CI/CD, having the ability to collect and aggregate real-time data about your systems health is critical.

There is substantial value in being alerted on site usage, processing times, different types of events, and exceptions. These real time health checks allow your engineers to properly assess issues and correct errors before things spiral out of control. If you fail to do so, your customers could become the reporters of your application issues. At that point your credibility is shot.

Monitoring has additional benefits as well. It provides valuable insights about user habits, and patterns. These feedback loops can be used to translate metrics from your applications and converted into real business value. Imagine being able to harness the power of making decisions based on real time usage reports.

The PMT is an excellent candidate to integrate into your pipeline process. There are many configurations and guides on implementing the PMT. If anything, plug it in and see how much it has to offer, you will not be disappointed on the sophisticated real-time data it provides.

One thing to note once you complete your PMT installation is that you are not required to perform the install multiple times when you execute your pipeline scripts. You can have a dedicated PMT setup that can be connected and configured to your newly spun up ColdFusion server.

Later in this section we will provide a sample .JAR file in how to automatically connect your ColdFusion builds with your PMT cluster.

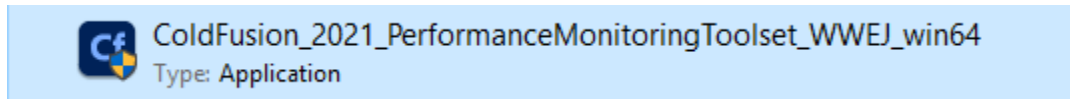
Install the ColdFusion [Performance Monitoring Toolset](#) ColdFusion 2021 Installation Instructions

Before installing the ColdFusion Performance Monitoring Toolset you will need to ensure that the package has been properly installed. In the ColdFusion Package Manager (CFPM) verify the **pmtagent** has been included in your build.

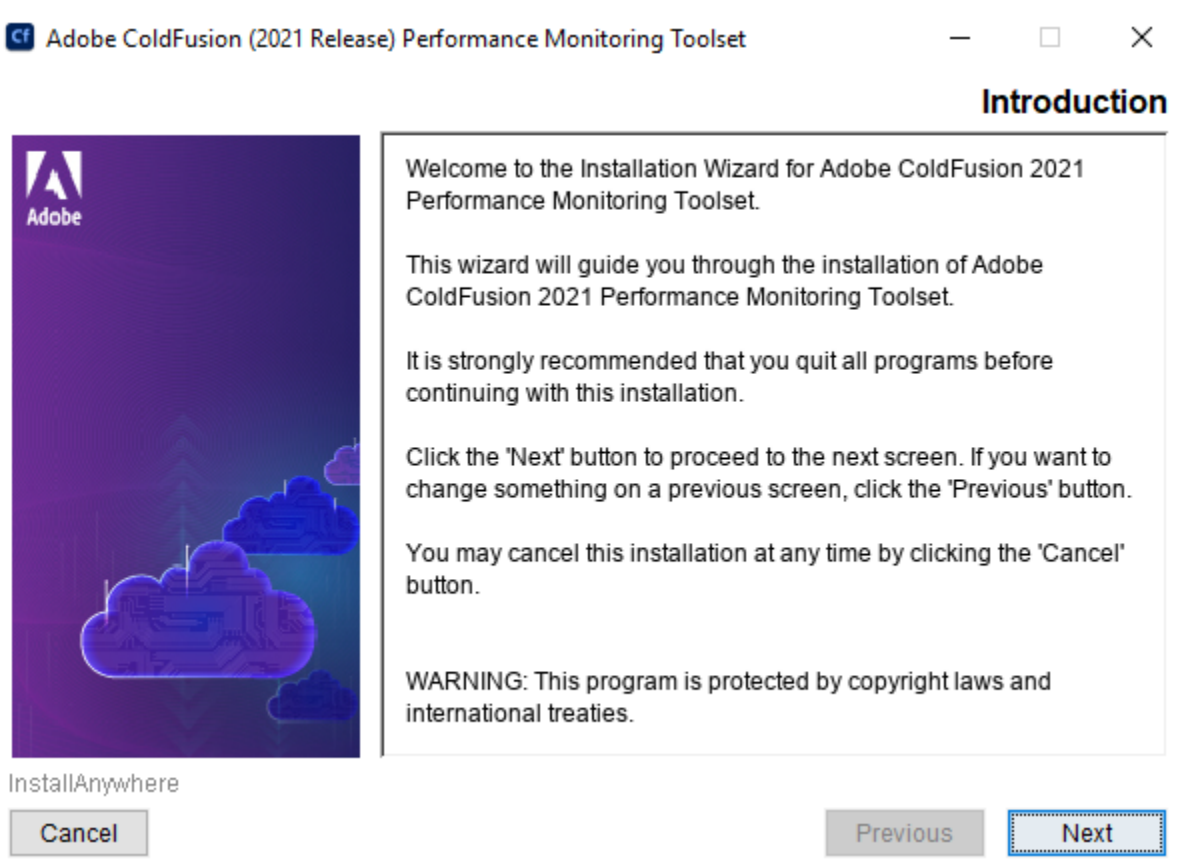
```
cfpm>install pmtagent_
```

Let's review the full Windows installation first. We will discuss the silent installation a bit later, along with code samples and the ability to include the installation in your Jenkins job.

Step 1: Download the latest .exe for the PMT. You can visit <https://www.adobe.com/support/coldfusion/downloads.html#cf2021pmt> to quickly find your preferred option.



Step 2: Launch the Windows installer by double clicking on the ColdFusion_2021_PerformanceMonitoringToolset_WWEJ_Win64.exe



Step 3: On the license agreement click Next

End User License Agreement



InstallAnywhere

Cancel

Installation and use of Adobe ColdFusion (2021 Release) Performance Monitoring Toolset requires acceptance of the following License Agreement:

ADOBE INC.

ADOBE COLDFUSION SOFTWARE

Software License Agreement (the "Agreement")

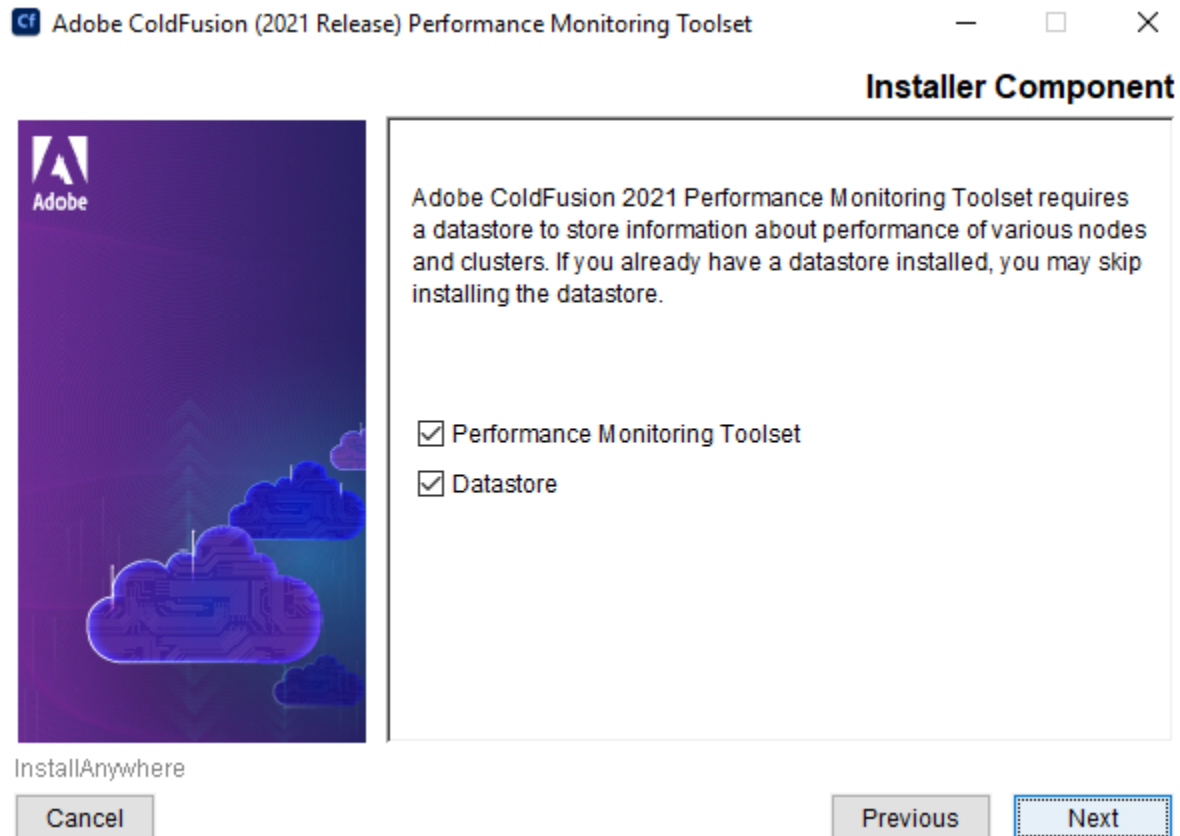
NOTICE TO USER: THIS AGREEMENT GOVERNS INSTALLATION AND USE OF THE ADOBE SOFTWARE DESCRIBED HEREIN. LICENSEE AGREES THAT THIS AGREEMENT IS LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY LICENSEE. BY CLICKING TO ACKNOWLEDGE AGREEMENT TO BE BOUND TO THE

☒ I accept the terms of the License Agreement

Previous

Next

Step 4: Choose the component to be installed. If you previously configured a Datastore there is no action to take at this step. Later in the process, you can specify the remote IP and port.



Step 6: Choose a location and click Next

Install Directory



InstallAnywhere

Cancel

Choose a folder for installing Adobe ColdFusion 2021 Performance Monitoring Toolset.

Install Directory

C:\ColdFusion2021PerformanceMonitoringToolset

Restore Default

Choose...

Previous

Next

Step 7: The default HTTP port for the PMT is 9101. You could change this to a different port if desired. Proceed and click Next.

Dashboard Configuration



Adobe ColdFusion 2021 Performance Monitoring Toolset will be configured to use 127.0.0.1:9101. You may specify a different IP address and port.

Hostname / IP

Port

InstallAnywhere

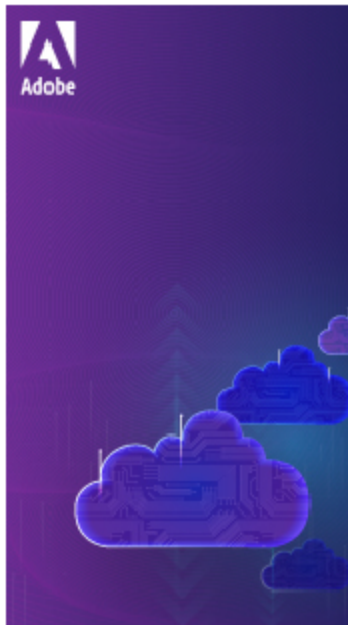
Cancel

Previous

Next

Step 8: Enter your administrator credentials these will be required for later access.

Administrator Credentials



InstallAnywhere

Cancel

Enter the username and password that you will use to restrict access to the Adobe ColdFusion 2021 Performance Monitoring Toolset Administrator.

Username

admin

Password

Confirm Password

Previous

Next

Step 9: Configure the Datastore. Install if required.

Datastore Configuration



InstallAnywhere

Cancel

Enter the hostname / IP and port of the machine where you want to install Datastore.

Note: You need to configure Datastore to start Adobe ColdFusion 2021 Performance Monitoring Toolset.

Hostname / IP

Port

Previous

Next

Step 10: Review the pre-installation summary and click Install

Pre-Installation Summary



Please review the following before continuing:

Install Folder:

C:\ColdFusion2021PerformanceMonitoringToolset

Host

127.0.0.1

Port

9101

Datastore Host

B702KC2

Datastore Port

9250

Disk Space Information (for Installation Target):

Required: 391.47 MegaBytes

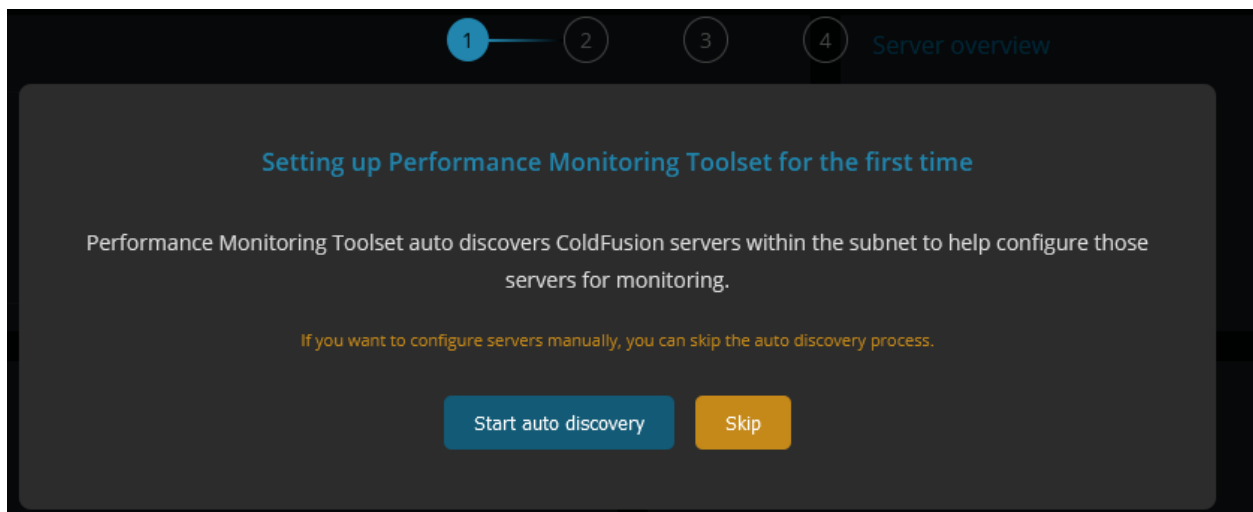
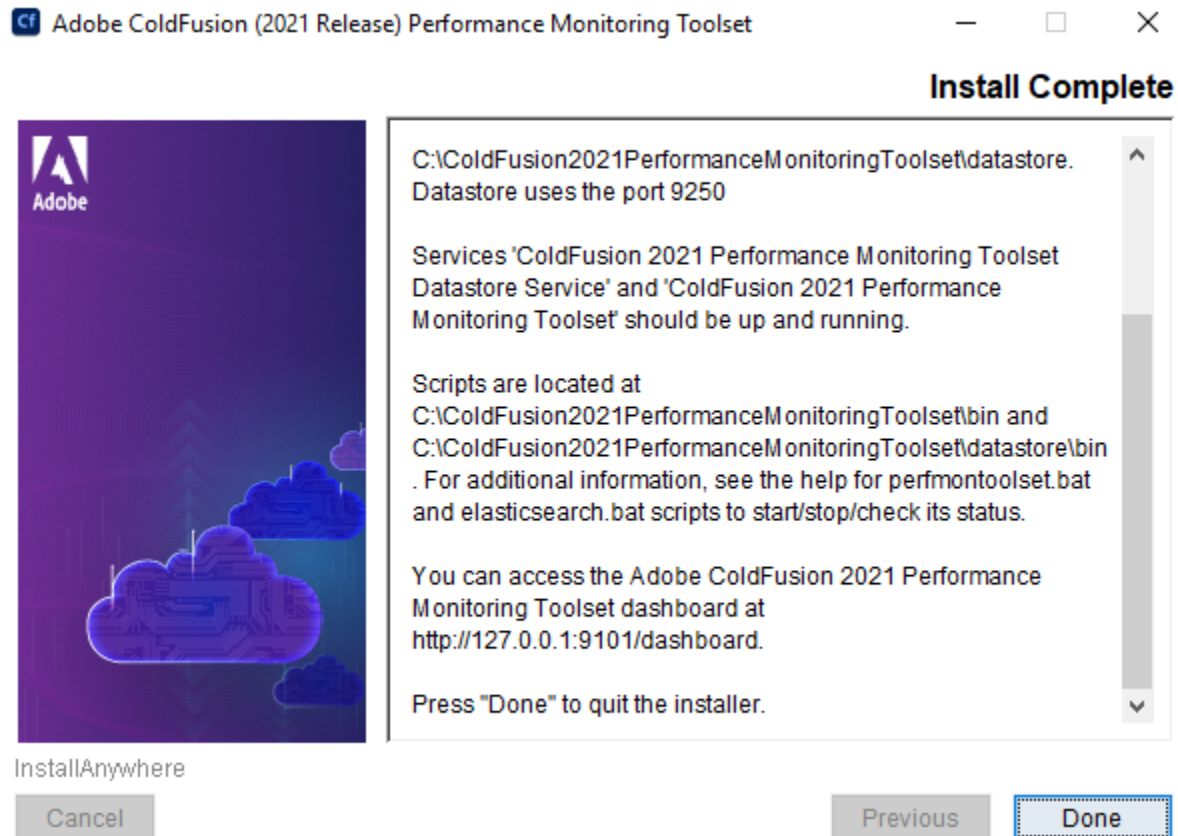
InstallAnywhere

Cancel

Previous

Install

Step 11: Upon successful completion of the install, the PMT should now launch in a browser window.



Install the ColdFusion [Performance Monitoring Toolset Silently](#)

Creating the properties file

The silent.properties file is an ASCII text file that defines the installation parameters. Specify the details as provided in the file. Select only the ones that apply to your installation type.

Saving the properties file

You can use a different name or save it in a different directory. If you require more than one installer properties file, give each file a descriptive name.

Running the installer

You can run the silent installer in one of the following ways:

1. Command line
2. Batch files or scripts

Below we have included the command line options when running the installer on different Operating Systems:

UNIX –

```
./ColdFusion_2021_PerformanceMonitoringToolset_WWEJ_Linux64.bin [-f propertiesFilePath]
```

Windows –

ColdFusion_2021_PerformanceMonitoringToolset_WWEJ_Win64.exe [-f propertiesFilePath]. For example, if you want to install Performance Monitoring Toolset silently, run the command, ColdFusion_2021_PerformanceMonitoringToolset_WWEJ_Win64.exe -f silent.properties.

Silent Installer Properties

Inform installer to start in silent mode.

```
INSTALLER_UI=SILENT
```

INSTALL_PMT is 1 if installer has to install Performance Monitoring Toolset component. Otherwise, INSTALL_PMT is 0.

If you install the Performance Monitoring Toolset component, it needs to connect to Datastore, which is either located in local (INSTALL_DATASTORE=1) or remote (INSTALL_DATASTORE=0).

INSTALL_PMT=1

Set the following variables only if INSTALL_PMT=1.

PMT_HOSTNAME must be either 127.0.0.1 or system hostname or system IP address

Default port is 9101. Port must be free.

Performance Monitoring Toolset service or process must be up and running to access Administrator Dashboard by URL http://PMT_HOSTNAME:PMT_PORT

PMT_HOSTNAME=127.0.0.1

PMT_PORT=9101

Performance Monitoring Toolset Administrator Dashboard login credentials.

USERNAME=admin

PASSWORD=Test#1234

PASSWORD_CONFIRM=Test#1234

Set INSTALL_DATASTORE to 1 if installer has to install DataStore. Otherwise INSTALL_DATASTORE is 0.

INSTALL_DATASTORE=1

If INSTALL_DATASTORE=1, the following two variables used for local installation. Otherwise, they are used to connect to remove the Datastore.

127.0.0.1 or 0.0.0.0 are not allowed as Hostname. Hostname should be either system hostname or IP address

The default port is 9200. Port must be free.

DATASTORE_HOSTNAME=HostNameorIPAddressoftheSystem

DATASTORE_PORT=9200

DATASTORE_PORT_CLUSTER=9300

OPTIONAL PROPERTIES

The location to install Performance Monitoring Toolset

Windows default location

USER_INSTALL_DIR=C:\\ColdFusion2018PerformanceMonitoringToolset

Mac OS X default location

USER_INSTALL_DIR=/Applications/ColdFusion2018PerformanceMonitoringToolset

Linux or Solaris default location

USER_INSTALL_DIR=/opt/coldfusion2018performancemonitoringtoolset

ONLY for Linux and Solaris platforms.

Make sure a runtime user exists in the system.

Setting "nobody" as a default username as it is the least privileged account in Linux and Solaris platforms.

RUNTIME_USER=nobody

Scripting your pipeline to connect your PMT and ColdFusion

Below you will find a sample java file that will allow your pipeline script to automatically connect your ColdFusion instances to your PMT.

STEP 1: Create a .java file, we will call it PMTService.java

STEP 2: Create the source code

```
package neo.qa.pmt;
```

```
import java.io.IOException;
```

```
import java.nio.charset.StandardCharsets;
```

```
import java.security.MessageDigest;
```

```
import java.security.NoSuchAlgorithmException;
```

```
import java.util.Base64;
```

```
import java.util.HashMap;
```

```
import java.util.Iterator;
```

```
import java.util.List;
```

```
import java.util.Map;
```

```
import org.apache.http.Header;
```

```
import org.apache.http.client.ClientProtocolException;
```

```
import org.apache.http.client.CookieStore;
```

```
import org.apache.http.client.config.RequestConfig;
```

```
import org.apache.http.client.methods.CloseableHttpResponse;
```

```
import org.apache.http.client.methods.HttpPost;
```

```
import org.apache.http.cookie.Cookie;
```

```
import org.apache.http.entity.ContentType;
```

```
import org.apache.http.entity.StringEntity;
```

```
import org.apache.http.impl.client.BasicCookieStore;
```

```
import org.apache.http.impl.client.CloseableHttpClient;
```

```
import org.apache.http.impl.client.HttpClients;
```

```
public class PMTService {
```

```
    private String PMT_SCHEME = "http";
```

```
    private String PMT_HOST = [PMT HOST IP];
```

```
    private String PMT_PORT = "9101";
```

```
    private static String PARTIAL_PMT_LOGIN_URL = "/pms/user/login";
```

```
    private static String PARTIAL_PMT_SINGLE_MONITOR_URL = "/pms/settings/discovery/singlemonitor";
```

```
    private static final int TIMEOUT = 60 * 1000;
```

```
    private static Header[] headers;
```

```
    private static Map<String, Object> responseHeaders = new HashMap<>();
```

```
    public PMTService() {    }
```

```
    public PMTService(String PMT_HOST, String PMT_PORT)
```



```

{
    this.PMT_HOST = PMT_HOST;
    this.PMT_PORT = PMT_PORT;
}

PMTService(String PMT_SCHEME, String PMT_HOST, String PMT_PORT)
{
    this(PMT_HOST, PMT_PORT);
    this.PMT_SCHEME = PMT_SCHEME;
}

public static void main(String[] args) {

    PMTService router = new PMTService();

    router.loginIntoPMT("admin", "admin");
    router.addCFServerToPMT("[server IP Address]", "8500", "cfusion", "73857f66-449f-4c72-82ee-
8513934571f2");
}

public void loginIntoPMT(String username, String password) {

    CloseableHttpClient httpclient;

    RequestConfig requestConfig = RequestConfig.custom().setConnectionRequestTimeout(TIMEOUT *
1000)

        .setConnectTimeout(TIMEOUT * 1000).setSocketTimeout(240 * 1000).build();

    CookieStore cookieStore = new BasicCookieStore();

    httpclient
    HttpClients.custom().setDefaultRequestConfig(requestConfig).setDefaultCookieStore(cookieStore).build();
=

```

```

String payload = getLoginPayload(username, password);
StringEntity entity = new StringEntity(payload, ContentType.APPLICATION_FORM_URLENCODED);

String PMT_LOGIN_URL = getBasePMTURL() + PARTIAL_PMT_LOGIN_URL;

HttpPost post = new HttpPost(PMT_LOGIN_URL);
post.setHeader("Content-Type", "application/json");
post.setEntity(entity);

try {
    CloseableHttpResponse response = httpClient.execute(post);

    headers = response.getAllHeaders();

    for (Header header : headers) {
        if (header.getName().equalsIgnoreCase("X-XSRF-Header"))
            responseHeaders.putIfAbsent(header.getName(), header.getValue());
    }

    List<Cookie> cookies = cookieStore.getCookies();

    for (Iterator iterator = cookies.iterator(); iterator.hasNext();) {
        Cookie cookie = (Cookie) iterator.next();
        responseHeaders.putIfAbsent(cookie.getName(), cookie);
    }

    System.out.println(response.getStatusLine().getStatusCode());
    // System.out.println(response);
} catch (ClientProtocolException e) {
    // TODO Auto-generated catch block

```

```

        e.printStackTrace();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public void addCFServerToPMT(String hostName, String port, String instanceName, String
monitoringSharedSecret) {

    CloseableHttpClient httpclient;

    RequestConfig requestConfig = RequestConfig.custom().setConnectionRequestTimeout(TIMEOUT *
1000)

        .setConnectTimeout(TIMEOUT * 1000).setSocketTimeout(240 * 1000).build();

    CookieStore cookieStore = new BasicCookieStore();

    httpclient =
    HttpClientBuilder.create().setDefaultRequestConfig(requestConfig).setDefaultCookieStore(cookieStore).build();

    String payload = getSingleMonitorPayload(hostName, port, instanceName,
monitoringSharedSecret);

    StringEntity entity = new StringEntity(payload, ContentType.APPLICATION_FORM_URLENCODED);

    String PMT_SINGLE_MONITOR_URL = getBasePMTURL() +
PARTIAL_PMT_SINGLE_MONITOR_URL;

    HttpPost post = new HttpPost(PMT_SINGLE_MONITOR_URL);
    post.setHeader("Content-Type", "application/json");
    post.setHeader("X-XSRF-Header", (String)responseHeaders.get("X-XSRF-Header"));
    post.setEntity(entity);

```

```
cookieStore.addCookie((Cookie)responseHeaders.get("PMS_COOKIE"));
cookieStore.addCookie((Cookie)responseHeaders.get("PMS_SESSION_COOKIE"));
```

```
try {
    CloseableHttpResponse response = httpClient.execute(post);

    System.out.println(response.getStatusLine().getStatusCode());
    // System.out.println(response);
} catch (ClientProtocolException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

```
}
```

```
private String getPasswdHash(String password) {
    MessageDigest messageDigest = null;
    try {
        messageDigest = MessageDigest.getInstance("SHA-256");
    } catch (NoSuchAlgorithmException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    byte[] digest = messageDigest.digest(password.getBytes(StandardCharsets.UTF_8));
    String hashed = Base64.getEncoder().encodeToString(digest);
    StringBuilder sb = new StringBuilder();
```

```

        for (int i = 0; i < digest.length; i++) {
            sb.append(Integer.toString((digest[i] & 0xff) + 0x100, 16).substring(1));
        }

        return sb.toString();
    }

    private String getLoginPayload(String username, String password) {
        String hashedPasswd = getPasswdHash(password);
        String payload = "{ \"username\": \"" + username + "\", \"password\": \"" + hashedPasswd + "\"
+ \"}";

        return payload;
    }

    private String getSingleMonitorPayload(String hostName, String port, String instanceName, String
monitoringSharedSecret) {

        if (null != hostName)
            hostName = hostName.trim();

        if (null != port)
            port = port.trim();

        if (null != instanceName)
            instanceName = instanceName.trim();

        String instanceld = hostName + ":" + instanceName + ":" + port;

        String payload = "{ \"manual\": true, \"instanceld\": \"" + instanceld + "\",
            + \"hostname\": \"" + hostName + "\", \"name\": \"Test\", \"port\": " +
Integer.parseInt(port) + ", \"j2ee\": false, "

```

```

+ "\"j2eeContext\": \"\", \"uuid\": \"\" + monitoringSharedSecret + "\", \"status\": \"error\"}";

        return payload;
    }

    private String getBasePMTURL() {

        return PMT_SCHEME + "://" + PMT_HOST + ":" + PMT_PORT;
    }
}

```

After completing your Java Class, it is time to plug it into the Pipeline.

Step 1: You will need to copy the Java file into the project directory workspace in Jenkins.

Possible path: Windows>System32>config>systemprofile>AppData>local>Jenkins

Step 2: It will be in the Jenkins working directory. Navigate to the .jenkins directory.


JS (C:) > Windows > System32 > config > systemprofile > AppData > Local > Jenkins

Name	Date modified	Type	Size
.jenkins	2/18/2021 9:01 PM	File folder	
war	2/18/2021 7:48 PM	File folder	
jenkins.pid	2/18/2021 7:47 PM	PID File	1 KB

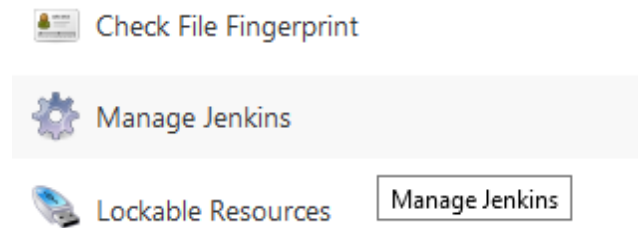
Step 3: Navigate into the workspace directory

workflow-libs	10/29/2020 9:26 PM	File folder	
workspace	2/18/2021 8:38 PM	File folder	
.lastStarted	2/18/2021 7:48 PM	LASTSTARTED File	0 KB
.owner	2/18/2021 8:51 PM	OWNER File	1 KB

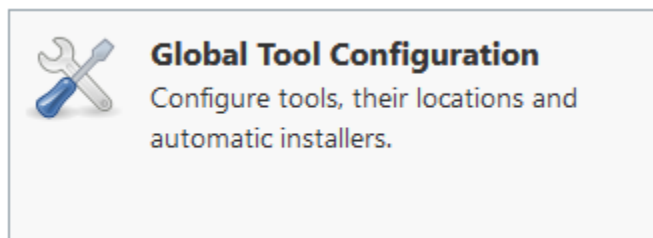
Step 4: Select your project and copy your Java File into the root.

:) > Windows > System32 > config > systemprofile > AppData > Local > Jenkins > .jenkins > workspace >				
Name	Date modified	Type	Size	
 ColdFusion2021 CI	2/18/2021 8:38 PM	File folder		

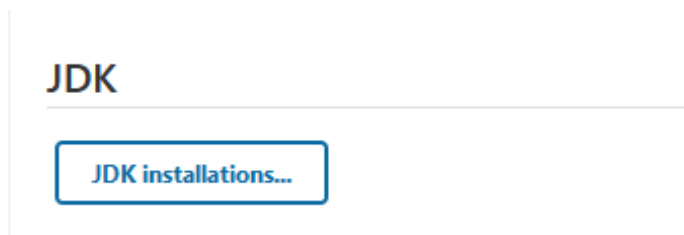
Step 5: Log into Jenkins and to set up your pipeline script and necessary configs. Select "Manage Jenkins"



Step 6: Select "Global Tool Configuration"



Step 7: Click on "JDK Installations"



Step 8: Select your JDK, Apply and Save

JDK

JDK installations

Add JDK

JDK

Name

JDK

JAVA_HOME

C:\Program Files\Java\jdk1.8.0_281\

Save

Apply

Step 9: Choose your current project in Jenkins and select the “Configure” option.

Dashboard > ColdFusion2021 CI >

Back to Dashboard

Status

Changes

Build Now

Configure

Pipeline ColdFusion2021 CI

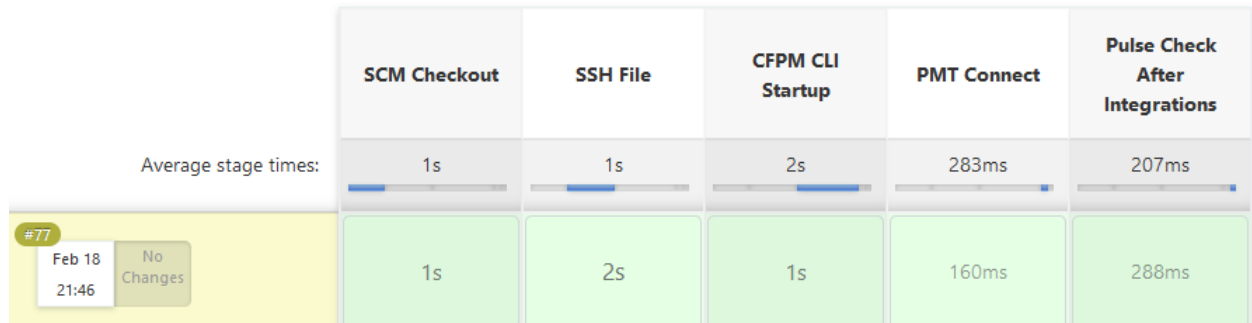
Recent Changes

Stage View

Step 10: Add the following script to your existing pipeline, in this case we will compile and execute. You can also execute a JAR file if you prefer.

```
stage('PMT Connect'){
    bat 'javac PMTService.java java PMTService'
}
```


Step 11: Execute your Pipeline and enjoy the comfort of knowing the PMT is now monitoring your ColdFusion applications.



Continuous Delivery and Deployments

Continuous Delivery

This brings us to the last part of the CI/CD pipeline. We have covered in detail the CI process. Continuous delivery (CD) is an extension of continuous integration. Critical decisions come into play at this point. Automated builds, executing test cases, and deploying files are just a few of the important dependencies required to reach a continuous delivery state.

The CD process encourages quality code that can be frequently delivered with predictability. Development teams can make changes to code and deliver small enhancements, without the angst of major manual deployments. With automation in place, the manual effort is removed from the equation, helping to repeat the same automated process with predictable results. In theory, Continuous Delivery is summarized as ensuring that code is ready to release at any moment. With a click of the button, code can be released.

Continuous Deployment

It is fair to say, especially when starting out, Continuous Delivery should be the initial goal. The next major step accelerates your process even further, and that would be the other CD, Continuous Deployment.

Continuous Deployment removes even more potential impediments in the pipeline process. Without the need for manual approvals, or a gate keeper, Continuous Deployment allows for automation through the entire pipeline to your production servers.

Development teams will be able to react to customer engagement in real time. Your business will be able to respond to changing markets and quickly deliver new features deploying code to production in a seamless and automated fashion.

Without the burden of a gate keeper needing to throw the switch for a deployment, you remove bottle necks from the deployment process. No longer will developers and business owners wait around for deployments.

It can come with a price, but if your acceptance tests are reliable, there should be little concern. Reinforcing this principle is the topic of shift-left testing. This type of automated testing can prove to be very beneficial for your pipeline. It is worth investigating to see how it can be implemented into your workflows.

Closing Remarks

In summary, ColdFusion 2018 and 2021 both offer several features to help move your organization into a CI/CD Pipeline. Whether you elect to use a CI server, some type of version control, or software to specifically test your builds, you have many options when proceeding with the implementation of a pipeline and ColdFusion based Applications.

ColdFusion continues to meet the needs of developers and organizations to fulfill the role of a rapid web-application development computing platform. The last two releases have been tremendous in keeping pace with the dynamic development industry.

Enhancements like the package manager, command line capabilities, performance monitoring, API Manager, enhanced security, and cloud integrations have enabled many organizations the ability to seamlessly move into the Continuous Integration pipelines.

Remember, start small, and explore where CI/CD can assist in moving your organization into new and exciting development opportunities. The benefits are tremendous, and ColdFusion, as always, will simplify your tasks in completing this amazing journey to improve your organizations development and deployment procedures.