

CODING Continuous Integration

Getting Started

Product Documentation





Copyright Notice

©2013-2024 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice

🔗 Tencent Cloud

All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Getting Started

Last updated : 2023-12-29 11:44:50

This document demonstrates how to use CODING Continuous Integration (CODING-CI) templates to deploy Node.js + Express + Docker applications.

Prerequisites

To use CODING-CI, you must activate the CODING DevOps service for your Tencent Cloud account.

Open Project

1. Log in to the CODING Console and click the team domain name to go to CODING.

2. Click



in the upper-right corner to open the project list page and click a project icon to open the corresponding project.3. Select **Continuous Integration** from the menu on the left.

Getting Started

1. Create build plan

After opening a project, select **Continuous Integration** on the left of the page, and then click **Build Plans** > **Create Build Plan**. With this plan, we will demonstrate the following based on Node.js + Express: Implement Auto Code Checkout > Unit Testing > Build Docker Image > Push to Docker Artifact Repository > Deploy to Remote Server (optional).

Overview	
Collaboration	
Repository	
Ode Scanner beta >	Welcome to Continuous Integration
∞ CI	
Build Job	Continuous Integration (CI) refers to a series of automated processes after code repository changes. Using appropriate CI tools and process orchestration
Build Node	can effectively improve development efficiency and implement rapid code Iteration and update.View Getting Started.
CD >	
Artifact Management	
差 Test Management >	
Document >	
	Create a build job Configure the build job Trigger and see results
	Choose the right template Fill in the necessary parameters Trigger builds and view steps and
	Create a build job
🌣 Settings <	

2. Select continuous integration template

Select a Node + Express + Docker continuous integration (CI) template.

Overview Collaboration ✓> Repository Occele Scanner beta	Select Build Job Template customize the build process. Build jobs are basic units in continuous integration. You can quickly create a build plan here. For more configurations, go to the build job details. View Help Documentation 12 All team template Programming Language Image Registry Artifact Repository Deploy Basics API Documentation Search by template keyword. Q
Cl Sulld Job Build Node	Java + Spring + Docker This template demonstrates the process of developing an application based on Java + Spring. The p Python + Flask + Docker This template demonstrates the process of developing an application based on Python + Flask. The
Artifact Management Test Management Document	Nodejs + Express + Docker This template demonstrates the process of developing an application based on Nodejs + Express. T
	Met Core + Docker This template demonstrates the process of developing an application based on .Net Core. The proc
	React + COS This template demonstrates the continuous integration build process based on React to achieve aut

3. Select code source

In the example project, we recommend you select **Sample Code** as the code source in the Code Repository field. Then, the system will automatically establish a sample code repository in your project. You can also select an existing code repository in a custom build plan.



Overview Collaboration Collaboration Code Scanner Beta > Code Scanner Beta >	Nodejs + Express + Docker Build Job Name * express-docker Build Process		👌 Template Details
Duris Jood Build Node A CD >> Artifact Management ▲ Test Management > Document >>	Code Repository Code Source Image: Code Name Image: Code Name	<pre>Jenkinsfile Preview pipeline { agent and the set of the</pre>	

4. Select CODING Docker artifact repository

After the build plan is completed, the system generates a build result. Here, you can select the CODING Docker artifact repository that the results are pushed to. If there is no artifact repository, you can quick create one.

			Docker Build Directory *	
🔂 Over	rview			<pre>stage('Build the image and push to the CODING Docker artifact repository') { stage /</pre>
	aboration		•	sh "docker build -t s{CODING DOCKER TMAGE NAME}:s{DOCKER TMAGE VERSION} -f s{DOCKERETLE PATH} s{DOCKER BUILD CONTEXT}"
	aboration			useCustomSteoPlugin(
Repo	ository		Docker Image Tag *	<pre>key: 'coding-public:artifact_docker_push', version: 'latest'.</pre>
🕲 Code	e Scanner beta >		Branch Name - Revision Number (\${GIT_LOCAL_B 🔻	params: [image "\${CODING DOCKER IMAGE NAME}.e{DOCKER IMAGE VERSION}"
00 CI	~			repo:"\$(DOCKE_REPO_WME)"
1				
Build	d Job	5	Push to CODING Docker Artifact Repository	}
Build	d Node		Docker Artifact Repository *	
				//stage("Deploy to remote service") {
A CD	>		Select an artifact repository.	// steps {
				// script {
Artifa	fact Management		Please search for Q	// def remoteConfig = [:]
				// remoteConfig.name = "my-remote-server"
📥 lest	t Management >	6	+ Create Artifact Repository	// remoteConfig.host = "\${REMOTE_HOST}"
			1 ordato Antihast Hopository	// remoteContig.port = "\${REMOTE_SSH_PORT}".toInteger()
Docu	ument >		Target Service Address *	// remoteContig.allowAnyHosts = true
				//
				// withereunitats() // sollierprivatekv/
				// credentialsId: "\${REMOTE CRED}".
				// keyFileVariable: "privateKeyFilePath"
			SSH Port *	
				// usernamePassword(
			22	<pre>// credentialsId: "\${CODING_ARTIFACTS_CREDENTIALS_ID}",</pre>
				<pre>// usernameVariable: 'CODING_DOCKER_REG_USERNAME',</pre>
A Sotti	inge //		SSH Username *	<pre>// passwordVariable: 'CODING_DOCKER_REG_PASSWORD'</pre>
Setti	inga ((oon oomano	
 Artifi Test Doct Setti 	fact Management t Management but Management	6	Please search for Q + Create Artifact Repository Target Service Address * Please enter Target Service Address. SSH Port * 22 SSH Username *	<pre>// stript(// def remoteConfig = [:] // remoteConfig.name = "my-remote-server" // remoteConfig.nost = "\${REMOTE_MOST}" // remoteConfig.nost = "\${REMOTE_SSH_PONT".toInteger() // remoteConfig.nost = "true // withCredentials([// sshUserPrivateKeyF // schUserPrivateKeyFilePath" // keyFileVariable: "privateKeyFilePath" // b, // usernamePassword(// credentials(1:"{CODIMG_ARTIFACTS_CREDENTIALS_ID}", usernamePassword(// credentials(1:"{CODIMG_DOCKER_REG_USERNAME", // passwordVariable: 'CODIMG_DOCKER_REG_USERNAME", // passwordVariable: 'CODIMG_DOCKER_REG_PASSWORD' //),</pre>

5. Enter remote server information (optional)

Enter the information of the remote server for deployment, including its IP address and port and SSH login credentials. After making sure this information is correct, wait for the build plan to be completed, after which the artifacts are sent to the remote server. You can preview the release result at a URL. If you do not need to deploy to a remote server, you can skip this step.



Overview Collaboration Repository	Docker Artifact Repository * Select an artifact repository.	<pre>//stage("Deploy to remote service") { // steps { // script { // script { // contectconfig = [:] // remoteConfig ,name = "mp-remote-server" // remoteConfig.host = "S[RDMTE_MOS]" // remoteConfig.host = "S[RDMTE_MOS]".toInteger() </pre>
Code Scanner beta >	6 Deploy to Remote Server Enable	// remoteConfig.allowAnyHosts = true //
00 CI 🗸	Target Service Address *	// withCredentials([// sshUserPrivateKev(
Build Job	Please enter Target Service Address.	<pre>// credentialsId: "\${REMDTE_CRED}", // keyFileVariable: "privateKeyFilePath"</pre>
Build Node	SSH Port *	//), // usernamePassword(// credentialsId: "\${CODING_ARTIFACTS_CREDENTIALS_ID}",
Artifact Management	22	// usernameVariable: 'CODING_DOCKER_REG_USERNAME', // passwordVariable: 'CODING_DOCKER_REG_PASSWORD' //)
👗 Test Management >	SSH Usemame *	//]) { // SH login username // constrained user were were were were
Document >	root	// remote.comig.user = "\$(MCMUL_USEK_UMME)" // // SHP private key file address // remoteComfig.ldenityFile = privateKeyFilePath
	SSH Login Credential *	// // // Please make sure there is a Docker environment in the remote environment
	Please select the credential.	// sshCommand(// remote: remoteConfig, // command: "docker lagin == s/2001UG_DOCKER_REG_UKERMANEL == s/2001UG_DOCKER_REG_RASEMMENT s/2001UG_DOCKER_REG_UKET!"
	Please search for Q	// commans overe cogin -u glovino_veren_negoveren_negoveren. // Stovino_veren_neg_resimonu/ Stovino_veren_neg_resi/ , // sude: true,
🌣 Settings 🔍	+ Enter the new credential and authorize the build job to	

After you click **Enter a new credential and authorize**, if you use an SSH private key to log in to the remote server, just click **Manually enter existing SSH private keys** for the entry method. After entering the key information, you can view it in **Project Settings** > **Developer Options** > **Credential Management**.



If you do not know how to log in to a remote server with an SSH private key, click **Automatically create an SSH key pair**. Then, you must manually set the public key in the <code>~ssh/authorized_keys</code> folder of the remote server.

6. Click create and view build result

Click **OK** to save the build plan. If you have selected **Trigger build after creation** the build plan will be executed immediately. During build plan execution, you can view build details in the list of build plan records.

Build Job		Shandhai 🕼 Status Badge	Scheduled Trigger	Q Cache 🏛 Settings	Build Now
My Stars System Source All Ungrouped More -		• shanghai	0 001000100 113301	Contra Contrago	
Trigger: All - Plan Source: Customize - Search Q	Only Me 🔵 Filter: All 🗵				
	Status	Trigger Info	Time Information	Quick View	Oper ation
express-docker O	Build succeeded.	Manually triggered by Tester #7 \$9 master - 0045b7e	8 minutes ago 54 seconds	" @ U &	
Sulld succeeded.	Automatically cancelled (duplic ate version number)	Manually triggered by Tester #6 \$° master - 0045b7e	🗎 8 minutes ago 🕓 –	~ 0 L	
Manually triggered by	Build succeeded.	Manually triggered by Tester #5 \$ master - 0045b7e	8 minutes ago 53 seconds	" @ L A	
	Build image and push to CODI NG Docker AR / Aborted by	Manually triggered by Tester #4 \$ master - 0045b7e	 9 minutes ago 44 seconds 	" @ L	
	Build succeeded.	Manually triggered by Tester #3 -o- 0045b7e	i0 minutes ago	" @ L &	
	1–7. Total: 7.			Entries per pa	.ge 15 👻 🚺

Click **Build Records** to view the operation statuses for each stage in the pipeline. You can also see the execution results and logs for the commands in each step.

Overview	Build Record #7 Build Process Build Snapshot Modificat	tion Record Test Report General Report Build Artifa	Check Out from Code Bepository (2 0 1 seconds - Full Sere
Collaboration Repository Code Scanner beta >	Build succeeded. Build succeeded. Iniger at 20 minutes ago, Duration 54 seconds	● ^{(sh} node-express-example ¹² master ^{(sh} 0045b7e ^{(ch}) Initial commit	7 [2022-02-23 162]:55] using GTT_SHI to set credentials 8 [2022-02-23 162]:55] using GTT_SHI to set credentials 8 [2022-02-23 162]:56] using If tech - tags - force - progress gittex_coding.net(spin):55/1/set/node-express-example.git +refs/med/shi refs/remote/origin/a/
oo Cl 🗸	Build Process		9 [2022-02-23 I6:21:57] > git config remote.orgign.url gitle.coding.net:ggl3567/test/node-express-example.git # timeout=10 10 [2022-02-23 I6:21:57] > git configadd remote.orgign.fetch +refs/heads/+:refs/remotes/origin/* # timeout=10
Build Node ♣ CD ▲ Artifact Management ▲ Test Management ▶ Document	Thek Out 1s	to run Shell script 10 s	 [202-20-23] B621373] - git config resolts-origin.url gitte-configured integration/test/indee-express-sample.git # insoltab [202-20-23] B621373] Fetting upstream changes from gitte-configured integration/test/indee-express-sample.git [202-20-23] B621373] and [Configured integration/test/indee-express-sample.git [202-20-23] B621373] and [Configured integration/test/indee-express-sample.git [202-20-23] B621373] and [Configured integration integration integration integration integration/test/indee-express-sample.git [202-20-23] B621373] and [Configured integration integratin integration integrat
🌣 Settings 🛛 «		-	

If the operation status of Step 5 is normal, you can view the artifact output URL in the build plan.

Overview Collaboration	Build Record #7 Build Process Build Snapshot Modification Re	acord Test Report General Report Build Artifa	Print Message (2) 26 seconds	Full Screen
 Repository Code Scanner beta > 	Build succeeded. Manually triggered by GP199513 Trigger at 25 minutes ago, Duration 54 seconds Initia		1 Build Success, Please go to http://18.131.116.1013000 for preview	
oo Cl V Build Job	Build Process			
Build Node A CD > Artifact Management ▲ Test Management > B Document >	10 s script 10 s Collect JUnit Test Re 1 s	auld the image and 1 s Deployment such to the CODING 1 s V To run Shell s V To run Shell s V Print Message		
🌣 Settings				

7. Modify remote server information

You have already configured a remote server address and SSH key in Step 5. To change this information, go to CI Plans > Settings > Variables and Caches.

í) M	Overview Collaboration	🔶 express-docker 🛽	Basic Info	Process Configuration T	rigger Rule	Variable and Cache	Notification
>	Repository Code Scanner beta >	Process Environment Variable Add the environment variable of the b	e)≣ Batch add stri puild job. When the b	ing type environment variables	+ Env Variable nent variable will s	erve as a default value of the	launch parameter.View the full help document. 🗹
00	CI 🗸	Variable Name	Category	Default Value	Operation		
	Build Job	DOCKER_IMAGE_NAME	String	nodejs-express-app	Z 8		
A	Build Node	DOCKER_BUILD_CONTEXT 📑	String		28		
	Artifact Management	DOCKERFILE_PATH	String	Dockerfile	28		
Ā	Test Management >	DOCKER_IMAGE_VERSION	String	\${GIT_LOCAL_BRANCH:-branch}-\${GI	๔ ⊗		
.8	Document >	DOCKER_REPO_NAME	String	test	๔ ⊗		
		Cache Directory 1. Enabling cache can avoid repetitive speed. 2. If an error occurs on your build cat 3. You are advised to enable cache fi	e download <mark>of</mark> the de che, reset the cache. or Maven, Gradle, an	pendency files in each build, greatly impro	oving the build		
٥	Settings 《				Reset Cache		

More Information

You can use CI plans to set custom CI build nodes.

Basic information

Cloud server and custom build node

Configuration process

Use the graphical interface to configure build processes



Build different types of artifacts and deliver them to the artifact repository

Trigger rules

Configure the CI trigger method

Variables and caches

Invoke security credentials and configure them in environment variables

Configure the project cache