Microsoft.GH-200.v2026-07-27.q90

Exam Code:	GH-200
Exam Name:	GitHub Actions
Certification Provider:	Microsoft
Free Question Number:	90
Version:	v2026-07-27
# of views:	103
# of Questions views:	991
https://www.dumpsfiles.com/files/Microsoft/GH-200/Microsoft.GH-200.v2026-07-27.q90	

NEW QUESTION: 1
As a developer, you are authoring a workflow that will deploy to both DevCloud and TestCloud resources.
Each cloud resource is accessed with a different deployment key. Which approach best allows you to use the same reusable workflow in separate jobs to target the different cloud resources?
A. Create repository secrets named DevCloud.DEPLOY_KEY and TestCloud.DEPLOY_KEY so that the reusable workflow parses the secrets by resource name.
B. Store the different keys in a DEPLOY_KEY environment secret in the DevCloud and TestCloud environments. Specify DEPLOY_KEY in the secrets section of the reusable workflow.
C. Use a marketplace action to conditionally parse the DEPLOY_KEY repository secret based on the cloud resource name.
D. Populate a DEPLOY_KEY repository secret with a JSON object containing DevCloud and TestCloud properties. Then specify DEPLOY_KEY.DevCloud in the secrets sections of the reusable workflow.
Answer: (SHOW ANSWER)
The best design is to use environment-scoped secrets with the same secret name, such as DEPLOY_KEY, in separate environments such as DevCloud and TestCloud. This lets the same reusable deployment logic reference one consistent secret name while the selected environment supplies the correct value. Option A is invalid because GitHub secret names should not be designed around dotted property access, and reusable workflows should not parse secret names dynamically. Option C introduces unnecessary marketplace dependency and weak secret handling. Option D is also wrong because GitHub secrets are opaque strings; ${{ secrets.DEPLOY_KEY.DevCloud }} is not valid secret property

access. Reusable workflows can define and receive named secrets through the secrets mapping.

NEW QUESTION: 2
Which workflow event is used to manually trigger a workflow run?
A. status
B. workflow_dispatch
C. workflow_run
D. create
Answer: B (LEAVE A REPLY)
Manually running a workflow
When a workflow is configured to run on the workflow_dispatch event, you can run the workflow using the Actions tab on GitHub, GitHub CLI, or the REST API.
Configuring a workflow to run manually
To run a workflow manually, the workflow must be configured to run on the workflow_dispatch event.
To trigger the workflow_dispatch event, your workflow must be in the default branch.
Reference:
https://docs.github.com/en/actions/how-tos/manage-workflow-runs/manually-run-a-workflow

NEW QUESTION: 3
As a developer, how can you identify a JavaScript action on GitHub?
A. The action's repository includes a js.yml file in the .github/workflows directory.
B. The action's repository name includes the keyword "JavaScript."
C. The action.yml metadata file has the runs.using value set to node16.
D. The action.yml metadata file references a package.json file.
Answer: D (LEAVE A REPLY)
Creating a JavaScript action, example
Prerequisites
Before you begin, you'll need to download Node.js and create a public GitHub repository.
* steps omitted *
From your terminal, initialize the directory with npm to generate a package.json file.
npm init -y
Adding actions toolkit packages
The actions toolkit is a collection of Node.js packages that allow you to quickly build JavaScript actions with more consistency.
When you commit and push your code, your updated repository should look like this:
hello-world-javascript-action/
─ action.yml
─ dist/

─ index.js
─ package.json
─ package-lock.json
─ README.md
─ rollup.config.js
─ src/
─ index.js
Reference:
https://docs.github.com/en/actions/tutorials/create-actions/create-a-javascript-action#creating-an- action-metadata-file

NEW QUESTION: 4
As a developer, your Actions workflow often reuses the same outputs or downloaded dependencies from one run to another. To cache dependencies for a job, you are using the GitHub cache action. Which input parameters are required for this action? (Each correct answer presents part of the solution. Choose two.)
A. path: the file path on the runner to cache or restore
B. dependency: the name and version of a package to cache or restore
C. key: the key created when saving a cache and the key used to search for a cache
D. restore-keys: the copy action key used with cache parameter to cache the data
E. cache-hit: the copy action key used with restore parameter to restore the data from the cache
F. ref: the ref name of the branch to access and restore a cache created
Answer: A,C (LEAVE A REPLY)
Input parameters for the cache action
[C] key: Required The key created when saving a cache and the key used to search for a cache.
It can be any combination of variables, context values, static strings, and functions. Keys have a maximum length of 512 characters, and keys longer than the maximum length will cause the action to fail.
[A] path: Required The path(s) on the runner to cache or restore.
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/dependency-caching

NEW QUESTION: 5
What are the most significant advantages of adding documentation while distributing custom actions? (Each answer presents a complete solution. Choose two.)
A. It provides an example of the action.
B. It creates a readme.md for the consuming workflow.
C. It shares the description of the action to the users.
D. It generates auto completion when using the action in a workflow.

Answer: B,C (LEAVE A REPLY)

NEW QUESTION: 6
How should you install the bats NPM package in your workflow?

A.


B.
```
jobs:
  steps:
    - run: npm install -g bats
```

C.
```
jobs:
  runs-on: ubuntu-latest
  - run: npm install -g bats
```

D.


Answer: D (LEAVE A REPLY)
The correct syntax includes specifying the job (example-job), the runner (ubuntu-latest), and the necessary step (npm install -g bats) within the workflow. This ensures that the package is installed properly during the execution of the job.

NEW QUESTION: 7
When creating and managing custom actions in an enterprise setting, which of the following is considered a best practice?
A. creating a separate repository for each action so that the version can be managed independently
B. creating a separate branch in application repositories that only contains the actions
C. creating a single repository for all custom actions so that the versions for each action are all the same
D. including custom actions that other teams need to reference in the same repository as application code
Answer: A (LEAVE A REPLY)
Creating a separate repository for each custom action allows you to manage the versioning independently for each action. This approach provides flexibility, as each action can be updated, tested, and versioned separately, avoiding potential conflicts or dependencies between different actions.

NEW QUESTION: 8
You need to use a specific version of an action. How should you use v1 of an actions/javascript-action repository in your workflow?
A. steps:
uses: actions/javascript-action/v1
B. steps:
uses: actions/javascript-action
version: v1
C. steps:
uses: actions/javascript-action-v1
D. steps:
uses: actions/javascript-action@v1
Answer: D (LEAVE A REPLY)
The correct syntax for referencing a versioned action is owner/repository@ref. Therefore, to use version v1 of actions/javascript-action, the workflow step must use actions/javascript-action@v1. The
@v1 portion is the ref, which can represent a tag, branch, or commit SHA depending on how the action publisher releases the action. Option A incorrectly treats the version as part of the repository path.
Option B invents a separate version: key, which is not how action versions are selected.
Option C changes the repository name and does not specify a ref. In GitHub Actions syntax, uses selects an action, and version selection is part of the action reference itself.
GitHub examples use the uses: owner
/action-name@ref form.

NEW QUESTION: 9
How can a workflow deploy mitigate the risk of multiple workflow runs that are deploying to a single cloud environment simultaneously? (Each correct answer presents part of the solution.
Choose two.)
A. Pass the mutex into the deployment job.
B. Set the concurrency in the deploymentjob to 1.
C. Specify a concurrency scope in the workflow.
D. Configure the mutex setting in the environment.
E. Reference the mutex in the task performing the deployment.
F. Specify a target environment in the deploymentjob.
Answer: (SHOW ANSWER)
[D] GitHub Actions now supports a concurrency key at both the workflow and job level that will ensure that only a single run or job is in progress.
concurrency

Use concurrency to ensure that only a single job or workflow using the same concurrency group will run at a time.
Example: Using concurrency and the default behavior
The default behavior of GitHub Actions is to allow multiple jobs or workflow runs to run concurrently. The concurrency keyword allows you to control the concurrency of workflow runs.
Reference:
https://github.blog/changelog/2021-04-19-github-actions-limit-workflow-run-or-job-concurrency/
https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-syntax#concurrency
https://github.com/marketplace/actions/actions-mutex

NEW QUESTION: 10
How should you print a debug message in your workflow?
A. echo "::debug::Set variable myVariable to true"
B. echo "Set variable MyVariable to true" >> $DEBUG_MESSAGE
C. echo "::add-mask::Set variable myVariable to true"
D. echo "debug_message=Set variable myVariable to true" >> &GITHUB_OUTPUT
Answer: (SHOW ANSWER)
Example: Setting a debug message
echo "::debug::Set the Octocat variable"
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-commands

NEW QUESTION: 11
Which YAML snippet triggers a GitHub Actions workflow for both push and pull request events?

A.
```
on:
  push:
    branches:
      - dev
  pull_request:
    branches:
      - main
```

B.
```
on:
  push:
    branches:
      - main
```

C.
```
on:
  push:
    branches:
      - main
  push:
    branches:
      - dev
```


D.
```
on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main
```


Answer: (SHOW ANSWER)

To trigger a GitHub Actions workflow on both push and pull request events, use the on key with a list of the events:

on:

push:

branches: ["main"]

pull_request:

branches: ["main"]

Key Details:

Events: You can list multiple events under on. This setup ensures the workflow runs whenever code is pushed to main or when a PR is opened/updated against main.

Activity Types: By default, pull_request triggers on opened, reopened, and synchronize (new commits to the PR).

Filtering: You can further refine these triggers by adding specific paths, tags, or different branches to avoid unnecessary runs Incorrect:

[Not A]

The push event is for the dev branch only.

[Not B]

Has only the push event.

[Not C]

Has only the push event.

Reference:

https://blog.devops.dev/automate-your-workflow-a-guide-to-ci-cd-with-github-actions-3f395d60ba69

NEW QUESTION: 12

What is the minimal syntax for declaring an output named foo for an action?

A.
```
outputs:
  foo:
    description: Some value
```

B.
```
outputs:
  - key: foo
    description: Some value
```

C.
```
outputs:
  foo:
    value: Some value
```

D.
```
outputs:
  - key: foo
    value: Some value
```

Answer: C (LEAVE A REPLY)

The correct minimal syntax for declaring an output in GitHub Actions is by using the foo key under outputs , and associating it with a value (in this case, Some value). This is the simplest form to define an output in a workflow or action.

NEW QUESTION: 13

What is a role of GitHub Marketplace when using GitHub Actions workflows?

A. GitHub Marketplace is used to manage secrets and environment variables for workflows.

B. GitHub Marketplace enables users to publish their own workflows and discover workflows created by others.

C. GitHub Marketplace provides a platform for hosting private workflows shared only within an organization.

D. GitHub Marketplace is used to run workflows on a specified schedule.

Answer: B (LEAVE A REPLY)

The GitHub Marketplace is primarily a platform for finding and sharing public actions, apps, and workflows with the entire GitHub community.

Incorrect:

[Not C]

The GitHub Marketplace is primarily a platform for finding and sharing public actions, apps, and workflows with the entire GitHub community. It is not the native platform for hosting workflows that are shared privately within an organization.

How to Share Workflows Privately

Rather than using the Marketplace, you can manage internal sharing through these methods:

Internal Repositories: You can store reusable workflows or actions in an internal repository. This makes them visible and usable by any other repository within the same organization or enterprise without making them public.
Private Repository Access: You can configure a private repository to allow access from other specific repositories or all repositories within your organization.
Reusable Workflows: By defining a workflow with the on: workflow_call trigger, you can reference it from other repositories using the owner/repo/path/to/workflow.yml@ref syntax.
Organizational Policies: Administrators can set policies at the organization or enterprise level to allow only internal actions, or a specific "allow list" of verified Marketplace actions.
Reference:
https://www.geeksforgeeks.org/git/what-is-github-marketplace/

NEW QUESTION: 14
What are the two types of environment protection rules you can configure? Each correct answer presents a complete solution. NOTE: Each correct answer is worth one point.
A. artifact storage
B. wait timer
C. branch protections
D. required reviewers
Answer: B,D (LEAVE A REPLY)
In GitHub Actions, you can configure four primary types of environment protection rules to control how and when deployments proceed to a specific environment.
These rules include:
[D] Required Reviewers: Specify up to six people or teams who must approve a deployment before it can proceed. Only one reviewer from the list needs to approve to unlock the deployment.
[B] Wait Timer: Set a mandatory delay (from 0 to 43,200 minutes) that must pass after a job is triggered before it can run.
Deployment Branches and Tags: Restrict deployments to only occur from specific branches or those matching certain tag patterns (e.g., main, release/*, or v*).
Custom Deployment Protection Rules: Enable third-party systems or automated tools (via GitHub Apps) to gate deployments based on external data, such as security scan results or ticket status.
Reference:
https://oneuptime.com/blog/post/2026-01-25-github-actions-environment-protection-rules/view

NEW QUESTION: 15
Disabling a workflow allows you to stop a workflow from being triggered without having to delete the file from the repo. In which scenarios would temporarily disabling a workflow be most useful?

(Each correct answer presents a complete solution. Choose two.)
A. A runner needs to have diagnostic logging enabled.
B. A workflow error produces too many, or wrong, requests, impacting external services negatively.
C. A workflow is configured to run on self-hosted runners.
D. A workflow sends requests to a service that is down.
E. A workflow needs to be changed from running on a schedule to a manual trigger.
Answer: (SHOW ANSWER)
Temporarily disabling a workflow can be useful in many scenarios. These are a few examples where disabling a workflow might be helpful:
[B] A workflow error that produces too many or wrong requests, impacting external services negatively.
A workflow that is not critical and is consuming too many minutes on your account.
[D] A workflow that sends requests to a service that is down.
Workflows on a forked repository that aren't needed (for example, scheduled workflows).
Reference:
https://docs.github.com/en/actions/how-tos/manage-workflow-runs/disable-and-enable-workflows

NEW QUESTION: 16
Your organization is managing secrets using GitHub encrypted secrets, including a secret named SuperSecret. As a developer, you need to create a version of that secret that contains a different value for use in a workflow that is scoped to a specific repository named MyRepo. How should you store the secret to access your specific version within your workflow?
A. Create MyRepo_SuperSecret in GitHub encrypted secrets to specify the scope to MyRepo.
B. Create and access SuperSecret from the secrets store in MyRepo.
C. Create a duplicate entry for SuperSecret in the encrypted secret store and specify MyRepo as the scope.
D. Create a file with the SuperSecret information in the .github/secrets folder in MyRepo.
Answer: C (LEAVE A REPLY)

NEW QUESTION: 17
You need to create new workflows to deploy to an unfamiliar cloud provider. What is the fastest and safest way to begin?
A. Create a custom action to wrap the cloud provider's CLI.
B. Search GitHub Marketplace for verified actions published by the cloud provider.
C. Use the actions/jenkins-plugin action to utilize an existing Jenkins plugin for the cloud provider.
D. Search GitHub Marketplace for actions created by GitHub.
E. Download the CLI for the cloud provider and review the associated documentation.
Answer: B (LEAVE A REPLY)
Searching the GitHub Marketplace for verified actions published by the cloud provider is the quickest and safest approach. Many cloud providers offer verified GitHub Actions that are maintained and optimized to interact with their services. These actions typically come with the correct configurations and best practices, allowing you to get started quickly without reinventing the wheel.
Note: About GitHub Marketplace for apps
GitHub Marketplace where you can share your apps with everyone.
GitHub Marketplace connects you to developers who want to extend and improve their GitHub workflows. You can list free and paid tools for developers to use in GitHub Marketplace. GitHub Marketplace offers developers two types of tools: GitHub Actions and Apps, and each tool requires different steps for adding it to GitHub Marketplace.
GitHub Actions
Anyone can publish an action in GitHub Marketplace. GitHub verifies some partner organizations and these are shown as verified creators.
Reference:
https://docs.github.com/en/enterprise-cloud@latest/apps/github-marketplace/github-marketplace- overview/about-github-marketplace-for-apps

NEW QUESTION: 18
As a developer, how can you identify a composite action on GitHub?
A. The action's repository name includes the keyword "composite."
B. The action's repository includes an init.sh file in the root directory.
C. The action's repository includes Dockerfile and package.json files.
D. The action.yml metadata file has the runs.using value set to composite.
Answer: D (LEAVE A REPLY)
runs.using for composite actions
Required: You must set this value to 'composite'.
Reference:

https://docs.github.com/en/actions/reference/workflows-and-actions/metadata-syntax

NEW QUESTION: 19
Which of the following is the best way for an enterprise to prevent certain marketplace actions from running?
A. Create a list of the actions that are restricted from being used as an enterprise policy. Every other action can be run.
B. It is not possible; if an action is in the marketplace, its use cannot be restricted.
C. Create a list that is maintained as a . yml file in a . github repository specified in the enterprise. Only these actions can be run.
D. Create a list of the actions that are allowed to run as an enterprise policy. Only these actions can be run.
Answer: (SHOW ANSWER)
The best way for an enterprise to control which GitHub Actions run is by creating a list of approved actions as an enterprise policy. This approach restricts workflows to only use the actions that are explicitly allowed, ensuring security and compliance within the organization.

NEW QUESTION: 20
Which configuration lines correctly set the workflow name in a GitHub Actions workflow file?
(Choose 2)
A. name: ${{ "release-flow-42" }}
B. name: ${{ 'release-flow-42' }}
C. run-name: release-flow-42
D. name: release-flow-42
Answer: B,D (LEAVE A REPLY)
The correct options are name: release-flow-42 and name: ${{ 'release-flow-42' }}. Both set the workflow name that appears in the Actions interface and on the workflow page.
The plain static name key with a simple string sets the workflow name clearly and predictably. It is the most straightforward way to define the workflow title.
The expression form with a single quoted string evaluates to the same literal value. GitHub Actions accepts a string result for the workflow name, so this produces a valid name while keeping consistency with other places where expressions are used.

NEW QUESTION: 21
You are reaching your organization's storage limit for GitHub artifacts and packages. What should you do to prevent the storage limit from being reached? (Choose two.)
A. Delete artifacts from the repositories manually
B. Disable branch protections in the repository.
C. Use self-hosted runners for all workflow runs.

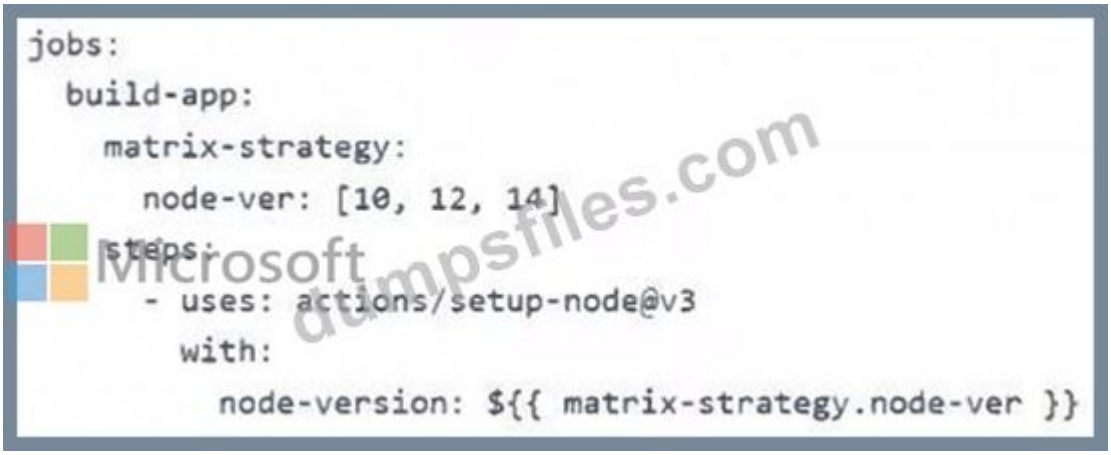
D. Configure the artifact and log retention period.
E. Configure the repo to use Git Large File Storage.
Answer: A,D (LEAVE A REPLY)
Deleting artifacts from repositories manually will free up storage space. Artifacts are typically stored for a limited time by default, but manual cleanup can help manage space. Configuring the artifact and log retention period allows you to control how long artifacts and logs are retained in your repository. By shortening the retention period, you can prevent unnecessary accumulation of data and manage storage more effectively.

NEW QUESTION: 22
Where should workflow files be stored to be triggered by events in a repository?
A. anywhere
B. Nowhere; they must be attached to an action in the GitHub user interface.
C. .github/workflows/
D. .workflows/
E. .github/actions/
Answer: B (LEAVE A REPLY)
You must store workflow files in the .github/workflows directory of your repository.
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-syntax

NEW QUESTION: 23
As a DevOps engineer, you are developing workflows to build an application. You have a requirement to create the build targeting multiple node versions. Which code block should you use to define the workflow?
A.

```
jobs:
  build-app:
    strategy:
      matrix:
        node-ver: [10, 12, 14]
    steps:
      - uses: actions/setup-node@v3
        with:
          node-version: ${{ matrix.node-ver }}
```

B.



```
jobs:
  build-app:
    matrix-strategy:
      node-ver: [10, 12, 14]
    steps:
      - uses: actions/setup-node@v3
        with:
          node-version: ${{ matrix-strategy.node-ver }}
```

C.

```
build-app:
  matrix:
    strategy:
      node-ver: [10, 12, 14]
  steps:
    - uses: actions/setup-node@v3
      with:
        node-version: ${{ matrix.node-ver }}
```

D.

```
jobs:
  build-app:
    strategy:
      matrix:
        node-ver: [10, 12, 14]
    steps:
      - uses: actions/setup-node@v3
        with:
          node-version: ${{ strategy.node-ver }}
```

Answer: A (LEAVE A REPLY)

Use keywords strategy and matrix in that order.

Last line should be node-version: ${{ matrix.node-ver}}

Reference:

https://docs.github.com/en/actions/how-tos/write-workflows/choose-what-workflows-do/run-job- variations

NEW QUESTION: 24

Which statement is true about using default environment variables?
A. The environment variables can be read in workflows using the ENV: variable_name syntax.
B. The environment variables created should be prefixed with GITHUB_ to ensure they can be accessed in workflows
C. The environment variables can be set in the defaults: sections of the workflow
D. The GITHUB_WORKSPACE environment variable should be used to access files from within the runner.
Answer: D (LEAVE A REPLY)
GITHUB_WORKSPACE is a default environment variable in GitHub Actions that points to the directory on the runner where your repository is checked out. This variable allows you to access files within your repository during the workflow.

NEW QUESTION: 25
You are a developer working on developing reusable workflows for your organization.
What keyword should be included as part of the reusable workflow event triggers?
A. check_run
B. workflow_run
C. workflow_call
D. pull_request
Answer: (SHOW ANSWER)
The workflow_call event is used to trigger a reusable workflow from another workflow. This allows you to create workflows that can be reused in multiple places within your organization, enabling better modularity and reducing duplication.

NEW QUESTION: 26
Which action type should be used to bundle a series of run steps into a reusable custom action?
A. Composite action
B. Bash script action
C. Docker container action
D. JavaScript action
Answer: A (LEAVE A REPLY)
A composite action allows you to bundle multiple steps into a single reusable action within a workflow. It is composed of multiple run steps or other actions and can be reused across workflows, making it the perfect choice for bundling a series of steps.

NEW QUESTION: 27
What is the most suitable action type for a custom action written in TypeScript?
A. Docker container action
B. Bash script action

C. composite run step
D. JavaScript action
Answer: D (LEAVE A REPLY)
Example: Build your custom GitHub Action
To build your action, run npm run bundle. This will compile the TypeScript code in the src/ directory and output the JavaScript code in the dist/ directory. The exact command for bundle is defined in the package.json file.
Note: About custom actions
Actions are individual tasks that you can combine to create jobs and customize your workflow. Y Types of action You can build Docker container, JavaScript, and composite actions.
Reference:
https://victoronsoftware.com/posts/typescript-github-action/
https://docs.github.com/en/actions/concepts/workflows-and-actions/custom-actions

NEW QUESTION: 28
Which workflow commands send information from the runner? (Choose two.)
A. reading from environment variables
B. setting a debug message
C. populating variables in a Dockerfile
D. setting output parameters
Answer: B,D (LEAVE A REPLY)
Setting a debug message using ::debug:: command sends a message to the logs, helping with troubleshooting and providing insight into the workflow run.
Setting output parameters using ::set-output sends data from a job step to subsequent steps or jobs, which can be used later in the workflow.

NEW QUESTION: 29
What is the smallest scope for an environment variable?
A. the workflow settings
B. a step
C. a job
D. the workflow env mapping
Answer: B (LEAVE A REPLY)
The smallest scope for an environment variable is within a step. Environment variables defined within a step are only accessible to that particular step, which makes it the smallest scope for a variable in a GitHub Actions workflow.

NEW QUESTION: 30

While writing a custom action, some behavior within the runner must be changed. Which workflow commands would set an error message in the runner's output? (Each correct answer presents a complete solution. Choose two.)

A. echo "::error file=main.py,line=10,col=15::There was an error"

B. echo "::error=There was an error::"

C. echo "::error::There was an error"

D. echo "::error message=There was an error::"

Answer: (SHOW ANSWER)

To set an error message in a GitHub Actions runner's output, you should use the command echo "::error::Your error message" in your workflow's run step. This workflow command creates an error annotation that will appear in the logs, and you can optionally include file and line number information, such as echo "::error file=app.js,line=5::Error message".

Reference: https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-commands

NEW QUESTION: 31

How can GitHub Actions encrypted secrets be used in if: conditionals within a workflow job?

A. Set the encrypted secret as a job-level environment variable and then reference the environment variable within the conditional statement.

B. Create a job dependency that exposes the encrypted secret as a job output, which can then be leveraged in a subsequent dependent job.

C. Use the secrets context within the conditional statement, e.g. ${{ secrets.MySuperSecret }}.

D. Use a workflow command to expose the encrypted secret via a step's output parameter and then use the step output in the job's if: conditional.

Answer: C (LEAVE A REPLY)

GitHub Actions encrypted secrets can be accessed in workflows using the secrets context. You can directly reference the secret within an if: conditional using ${{ secrets.MySuperSecret }} to determine whether a job or step should run based on the secret's value.

NEW QUESTION: 32
As a developer, you want to review the step that caused a workflow failure and the failed step's build logs. First navigate to the main page of the repository on GitHub. Which section contains the step failure information?
A. Insights
B. Code
C. Actions
D. Pull requests
E. Issues
Answer: C (LEAVE A REPLY)
The Actions tab on the main page of the repository is where you can find detailed information about the workflow runs, including step failures and build logs. You can review the status of each job and step within the workflow, see the failure messages, and access logs for debugging.

NEW QUESTION: 33
Based on the YAML below, which two statements are correct? (Choose two.)

```
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-node@v1
        with:
          node-version: 12
      - run: npm ci
      - run: npm test

  publish-npm:
    needs: build
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-node@v1
        with:
          node-version: 12
      - run: npm ci
      - uses: JS-DevTools/npm-publish@v1
        with:
          token: ${{ secrets.NPM_TOKEN }}
```


A. This workflow will publish a package to an npm registry.
B. This workflow will publish a package to GitHub Packages.

C. This workflow file is using a matrix strategy.
D. The workflow job publish-npm will only run after the build job passes.
Answer: A,D (LEAVE A REPLY)
The publish-npm job includes the JS-DevTools/npm-publish action, which is used to publish an npm package to an npm registry.
The publish-npm job has the needs: build directive, meaning it will only run after the build job successfully completes.

NEW QUESTION: 34
Which code version is the default for scheduled triggered workflows?
A. Scheduled workflows run on the:
B. latest commit and branch on which the workflow was triggered,
C. latest commit from the branch named schedule,
D. latest commit from the branch named main,
E. specified commit and branch from the workflow YAML file,
F. latest commit on the default or base branch
Answer: A (LEAVE A REPLY)
Scheduled workflows in GitHub Actions are triggered at specified times, and they run on the latest commit of the branch that triggers the workflow. This means the workflow will run on the most recent commit on the branch that was active at the time the scheduled event occurs.

NEW QUESTION: 35
Which workflow commands send information from the runner? (Each correct answer presents a complete solution. Choose two.)
A. reading from environment variables
B. setting a debug message
C. populating variables in a Dockerfile
D. setting output parameters
Answer: B,D (LEAVE A REPLY)
[B] Setting a debug message
Prints a debug message to the log. You must create a secret named ACTIONS_STEP_DEBUG with the value true to see the debug messages set by this command in the log.
::debug::{message}
Example: Setting a debug message
echo "::debug::Set the Octocat variable"
[D] Setting an output parameter
Sets a step's output parameter. Note that the step will need an id to be defined to later retrieve the output value.
echo "{name}={value}" >> "$GITHUB_OUTPUT"

Example of setting an output parameter
This example demonstrates how to set the SELECTED_COLOR output parameter and later retrieve it:
- name: Set color
id: color-selector
run: echo "SELECTED_COLOR=green" >> "$GITHUB_OUTPUT"
- name: Get color
env:
SELECTED_COLOR: ${{ steps.color-selector.outputs.SELECTED_COLOR }}
run: echo "The selected color is $SELECTED_COLOR"
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-commands

NEW QUESTION: 36
Scheduled workflows run on the:
A. latest commit and branch on which the workflow was triggered,
B. latest commit from the branch named schedule,
C. latest commit from the branch named main,
D. specified commit and branch from the workflow YAML file,
E. latest commit on the default or base branch
Answer: (SHOW ANSWER)
Scheduled workflows in GitHub Actions are triggered at specified times, and they run on the latest commit of the branch that triggers the workflow. This means the workflow will run on the most recent commit on the branch that was active at the time the scheduled event occurs.

NEW QUESTION: 37
You create a self-hosted runner labeled as runner1.
You need to ensure that a GitHub Actions workflow job runs only on runner1.
Which YAML statement should you use?
A. runs-on: [self-hosted, linux-xlarge]
B. runs-on: - self-hosted - ubuntu-latest
C. runs-on: [self-hosted, runner1]
D. runs-on: self-hosted
Answer: C (LEAVE A REPLY)
To target a specific self-hosted runner, the job must use runs-on with labels that match that runner. A self-hosted runner normally has the self-hosted label plus default labels and any custom labels you assign. Since the runner is labeled runner1, the workflow should specify both self-hosted and runner1:
runs-on: [self-hosted, runner1]. GitHub Actions will then route the job only to a self-hosted runner matching all specified labels. Option A targets a different custom label. Option B is

malformed and also mixes a self-hosted label with a GitHub-hosted runner label. Option D targets any self-hosted runner, not specifically runner1. GitHub documents that an array of runs-on labels requires a runner matching all listed labels.

NEW QUESTION: 38
You are a DevOps engineer working on a custom action. You want to conditionally run a script at the start of the action, before the main entrypoint. Which code block should be used to define the metadata file for your custom action?
A. runs:
using: ' node16 '
start: ' start.js '
start-if: github.event_name == ' push '
main: ' index.js '
B. runs:
using: ' node16 '
pre: ' start.js '
pre-if: github.event_name == ' push '
main: ' index.js '
C. runs:
using: ' node16 '
pre-if: github.event_name == ' push ' then ' start.js '
main: ' index.js '
D. runs:
using: ' node16 '
before: ' start.js '
before-if: github.event_name == ' push '
main: ' index.js '
Answer: B (LEAVE A REPLY)
For JavaScript custom actions, the metadata file supports lifecycle scripts through the runs section. The correct key for a script that runs before the main action entrypoint is pre, and the correct conditional key for controlling whether that pre-script runs is pre-if. Therefore, option B is the only valid metadata structure.
GitHub Actions does not use start, start-if, before, or before-if as action metadata keys. Option C is also invalid because pre-if only defines a condition; it cannot contain both a condition and a script reference in one line. In GitHub Actions exam terms, this tests custom action metadata, JavaScript action lifecycle hooks, and conditional execution before the main action logic.

NEW QUESTION: 39
You have exactly one Windows x64 self-hosted runner, and it is configured with custom tools. Which syntax could you use in the workflow to target that runner?

A. self-hosted: [windows-x64]
B. runs-on: [self-hosted, windows, x64]
C. runs-on: windows-latest
D. self-hosted: [windows, x64]
Answer: (SHOW ANSWER)
The runs-on keyword allows you to specify the operating system and other labels for the runner. By specifying self-hosted, windows, and x64, you are targeting a self-hosted Windows runner that matches these criteria, which aligns with the custom configuration of your self-hosted runner.

NEW QUESTION: 40
As a DevOps engineer, you need to define a deployment workflow that runs after the build workflow has successfully completed. Without modifying the build workflow, which trigger should you define in the deployment workflow?
A. repository_dispatch
B. workflow_dispatch
C. workflow_exec
D. workflow_run
Answer: (SHOW ANSWER)
A deployment workflow can be started after a build workflow has finished by using the workflow_run event in the deployment workflow's on trigger. You must specify the name of the build workflow you want to trigger on and use an if condition to ensure the deployment workflow only runs if the build workflow successfully completes.
Here's how to set it up:
In your deployment workflow file: (e.g., deploy.yml), define the on: trigger.
Use the workflow_run event: within the on: trigger.
Specify the build workflow: by its name.
Add a conditional if statement: to the workflow to check the conclusion of the workflow_run event, ensuring it equals 'success'.
Reference:
https://docs.github.com/actions/learn-github-actions/events-that-trigger-workflows

NEW QUESTION: 41
As a developer, you want to run a workflow from the Actions tab in GitHub. Which YAML snippet should you use to match the interface in this image?

A.

```
on:
  workflow_dispatch:
    inputs:
      test_suite:
        description: Test suite
        type: choice
        options:
          - functional
          - regression
```

B.

```
on:
  workflow_run:
    inputs:
      test_suite:
        description: Test suite
        type: string
        options:
          - functional
          - regression
```

C.

```
on:
  workflow_dispatch:
    inputs:
      test_suite:
        description: Test suite
        type: choice
        value: functional
        options:
          - regression
```

D.

```
on:
  workflow_run:
    inputs:
      test_suite:
        description: Test suite
        type: choice
        options:
        - functional
        - regression
```

Answer: C (LEAVE A REPLY)

The first image shows a workflow trigger with an option for the test suite, and the chosen YAML configuration matches this interface. Specifically, the test suite input is defined with type: choice and includes the option value: functional , which aligns with the visible UI elements in the first image.

NEW QUESTION: 42

As a developer, your Actions workflow often reuses the same outputs or downloaded dependencies from one run to another. To cache dependencies for a job, you are using the GitHub cache action. Which input parameters are required for this action? (Choose two.)

A. dependency: the name and version of a package to cache or restore

B. key: the key created when saving a cache and the key used to search for a cache

C. cache-hit: the copy action key used with restore parameter to restore the data from the cache

D. path: the file path on the runner to cache or restore

E. ref: the ref name of the branch to access and restore a cache created

F. restore-keys: the copy action key used with cache parameter to cache the data

Answer: B,D (LEAVE A REPLY)

The key is required because it uniquely identifies the cache. It is used to store and retrieve cached data. When creating or restoring a cache, you need to define a key that will be used to identify the cache.

The path is the file path on the runner that you want to cache. This is where the cached files or dependencies are located and should be specified to tell GitHub where to store or retrieve the cache from.

NEW QUESTION: 43

What can be used to set a failed status of an action from its code?

A. @actions/github toolkit

B. JavaScript dist/ folder

C. Dockerfile CMD

D. a non-zero exit code

E. output variable

F. composite run step

Answer: D (LEAVE A REPLY)
A non-zero exit code is used to set the status of an action to "failed" in GitHub Actions. When the action's script or code exits with a non-zero status, it indicates failure, and GitHub will mark the action as failed.

NEW QUESTION: 44
Which files are required for a Docker container action in addition to the source code? (Each correct answer presents a partial solution. Choose two.)
A. action.yml
B. Actionfile
C. metadata.yml
D. Dockerfile
Answer: A,D (LEAVE A REPLY)
Creating a Docker container action
Create a dockerfile
Dockerfile support for GitHub Actions
When creating a Dockerfile for a Docker container action, you should be aware of how some Docker instructions interact with GitHub Actions and an action's metadata file.
Creating an action metadata file
Create a new action.yml file.
Reference:
https://docs.github.com/en/actions/tutorials/use-containerized-services/create-a-docker-container- action
https://docs.github.com/en/actions/reference/workflows-and-actions/dockerfile-support

NEW QUESTION: 45
What is the right method to ensure users approve a workflow before the next step proceeds?
A. creating a branch protection rule and only allow certain users access
B. granting users workflow approval permissions
C. adding users as required reviewers for an environment
D. granting users repository approval permission
Answer: (SHOW ANSWER)
GitHub Actions allows you to configure environment protection rules, where you can require specific users or teams to approve the deployment before the workflow proceeds to the next step.
This ensures that the required reviewers approve the workflow before any sensitive actions (such as deployment) occur.
Note: Managing environments for deployment

You can create environments and secure those environments with deployment protection rules. A job that references an environment must follow any protection rules for the environment before running or accessing the environment's secrets.
Optionally, you can specify people or teams that must approve workflow jobs that use this environment.
Select Required reviewers.
Enter up to 6 people or teams. Only one of the required reviewers needs to approve the job for it to proceed.
Optionally, to prevent users from approving workflows runs that they triggered, select Prevent self-review.
Click Save protection rules.
Reference:
https://docs.github.com/en/actions/how-tos/deploy/configure-and-manage-deployments/manage- environments

NEW QUESTION: 46
How can a workflow deploy mitigate the risk of multiple workflow runs that are deploying to a single cloud environment simultaneously? (Each correct answer presents part of the solution.
Choose two.)
A. Reference the mutex in the task performing the deployment.
B. Set the concurrency in the deploymentjob to 1.
C. Specify a target environment in the deploymentjob.
D. Specify a concurrency scope in the workflow.
E. Configure the mutex setting in the environment.
F. Pass the mutex into the deployment job.
Answer: C,D (LEAVE A REPLY)
[D] GitHub Actions now supports a concurrency key at both the workflow and job level that will ensure that only a single run or job is in progress.
concurrency
Use concurrency to ensure that only a single job or workflow using the same concurrency group will run at a time.
Example: Using concurrency and the default behavior
The default behavior of GitHub Actions is to allow multiple jobs or workflow runs to run concurrently. The concurrency keyword allows you to control the concurrency of workflow runs.
Reference:
https://github.blog/changelog/2021-04-19-github-actions-limit-workflow-run-or-job-concurrency/
https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-syntax#concurrency

https://github.com/marketplace/actions/actions-mutex

NEW QUESTION: 47
As a developer, your self-hosted runner sometimes looses connection while running jobs. How should you troubleshoot the issue affecting your self-hosted runner?
A. Set the DEBUG environment variable to true before starting the self-hosted runner to produce more verbose console output.
B. Locate the self-hosted runner in your repository's settings page and download its log archive.
C. Access the self-hosted runner's installation directory and look for log files in the _diag folder.
D. Start the self-hosted runner with the --debug flag to produce more verbose console output.
Answer: C (LEAVE A REPLY)
When troubleshooting a self-hosted runner, you can access the _diag folder located in the self-hosted runner's installation directory. This folder contains diagnostic logs that can help you identify the root cause of issues, such as connection problems.

NEW QUESTION: 48
Which choices represent best practices for publishing actions so that they can be consumed reliably? (Each correct answer presents a complete solution. Choose two.)
A. default branch
B. commit SHA
C. repo name
D. tag
E. organization name
Answer: B,D (LEAVE A REPLY)
Security practices for writing workflows and using GitHub Actions features.
You can help mitigate this risk by following these good practices:
[B] Pin actions to a full-length commit SHA
Pinning an action to a full-length commit SHA is currently the only way to use an action as an immutable release. Pinning to a particular SHA helps mitigate the risk of a bad actor

adding a backdoor to the action's repository, as they would need to generate a SHA-1 collision for a valid Git object payload. When selecting a SHA, you should verify it is from the action's repository and not a repository fork.
[D] Pin actions to a tag only if you trust the creator
Although pinning to a commit SHA is the most secure option, specifying a tag is more convenient and is widely used. If you'd like to specify a tag, then be sure that you trust the action's creators.
The 'Verified creator' badge on GitHub Marketplace is a useful signal, as it indicates that the action was written by a team whose identity has been verified by GitHub. Note that there is risk to this approach even if you trust the author, because a tag can be moved or deleted if a bad actor gains access to the repository storing the action.
Reference:
https://docs.github.com/en/actions/reference/security/secure-use

NEW QUESTION: 49
Which action type should be used to bundle a series of run steps into a reusable custom action?
A. Composite action
B. Bash script action
C. Docker container action
D. JavaScript action
Answer: (SHOW ANSWER)
Reusable workflows versus composite actions
Reusable workflows and composite actions both help you avoid duplicating workflow content.
Whereas reusable workflows allow you to reuse an entire workflow, with multiple jobs and steps, composite actions combine multiple steps that you can then run within a job step, just like any other action.
A composite action allows you to bundle multiple steps into a single reusable action within a workflow. It is composed of multiple run steps or other actions and can be reused across workflows, making it the perfect choice for bundling a series of steps.
Reference:
https://docs.github.com/en/actions/concepts/workflows-and-actions/reusable-workflows

NEW QUESTION: 50
Which of the following scenarios would require the use of self-hosted runners instead of GitHub-hosted runners?
A. exceeding 50,000 monthly minutes of build time
B. using Docker containers as part of the workflow
C. running more than the three concurrent workflows supported by GitHub-hosted runners
D. performing builds on macOS

E. using specialized hardware configurations required for workflows

Answer: C,E (LEAVE A REPLY)

GitHub-hosted runners have a limit on the number of concurrent workflows (typically 20 for free-tier accounts and 5 for enterprise). If your organization needs to run more workflows simultaneously, you would need to use self-hosted runners to increase the available concurrency.

Self-hosted runners allow you to configure specialized hardware or software setups that are necessary for certain workflows. GitHub-hosted runners may not have access to custom hardware configurations like GPUs or other specialized resources, so self-hosted runners are required in such cases.

NEW QUESTION: 51

What is the most secure way to store sensitive information in GitHub Actions workflows?

A. From the GitHub repository settings, store sensitive data as unencrypted text.

B. Use environment variables in the workflow and store sensitive data directly in the variables.

C. Store the sensitive information as plain text in the workflow YAML file.

D. Use GitHub Enterprise Managed User (OIDC) integration to dynamically fetch secrets from a third- party secrets manager.

Answer: (SHOW ANSWER)

NEW QUESTION: 52

In the following workflow file, line 5 interprets lines 3 and 4 as Python. Which of the following is a valid option to complete line 5?

```
1 steps:
2 - run: |
3 import os
4 print(os.environ[ ' PATH ' ])
5
```

A. with: python

B. shell: bash

C. working-directory: .github/python

D. shell: python

Answer: D (LEAVE A REPLY)

The run keyword executes commands using the default shell unless a different shell is specified. Since the commands shown are Python statements, the step must define shell: python so GitHub Actions interprets the script using Python instead of Bash, PowerShell, or another default shell. Option D is therefore correct.

Option A is invalid because with: is used to pass inputs to actions, not to define the interpreter for a run command. Option B would execute the text as Bash and fail because import os is not Bash syntax. Option C only changes the working directory and does not

change the shell. This exam topic checks workflow syntax, shell selection, and step-level command execution.

NEW QUESTION: 53
Your organization needs to simplify reusing and maintaining automation in your GitHub Enterprise Cloud. Which components can be directly reused across all repositories in an organization? (Choose three.)
A. self-hosted runners
B. actions stored m private repositories in the organization
C. encrypted secrets
D. custom Docker actions stored in GitHub Container Registry
E. actions stored in an organizational partition in the GitHub Marketplace
F. workflow templates
Answer: B,C,F (LEAVE A REPLY)
Actions stored in private repositories within the organization can be reused across all repositories by referencing them in workflows. This ensures a centralized way of maintaining custom actions.
Encrypted secrets can be accessed across repositories in the same organization, making it easy to store sensitive data (like API keys or tokens) securely while allowing multiple workflows to access them.
Workflow templates allow you to create reusable templates for workflows that can be shared across repositories within the organization. This makes it easier to standardize processes and automate them across multiple projects.

NEW QUESTION: 54
As a DevOps engineer, you are trying to leverage an organization secret in a repo. The value received in the workflow is not the same as that set in the secret. What is the most likely reason for the difference?
A. There is a different value specified at the rego level.
B. There is a different value specified at the workflow level.
C. The Codespace secret doesn't match the expected value.
D. The Encrypt Secret setting was not configured for the secret.
E. There is a different value specified at the enterprise level.
Answer: (SHOW ANSWER)
GitHub secrets are defined at different levels: organization, repository, and sometimes at the workflow level. If a secret is defined at both the organization level and the repository level, the repository-level secret will take precedence. So, if the value of the secret differs between these levels, the workflow will use the value from the repository level instead of the organization level.

NEW QUESTION: 55

What are the two types of environment protection rules you can configure? (Choose two.)
A. required reviewers
B. branch protections
C. wait timer
D. artifact storage
Answer: A,C (LEAVE A REPLY)
Required reviewers is a protection rule where you can specify that certain individuals or teams must review and approve the workflow run before it can proceed. This is used to enforce approvals before certain steps or environments are accessed.
Wait timer is a protection rule that introduces a delay before a workflow can proceed to the next stage. This is useful for adding time-based constraints to the deployment process or ensuring that certain conditions are met before a workflow continues.

NEW QUESTION: 56
For which type of GitHub Actions workflows should you use a self-hosted runner?
A. workflows that require access to private internal network resources
B. workflows that have no specific environment configurations and must be triggered on a GitHub- hosted infrastructure
C. short-lived workflows that do NOT require custom dependencies
D. workflows that require the latest operating system and preinstalled software versions
Answer: A (LEAVE A REPLY)
Self-hosted Runners: These run on your own infrastructure (on-premises or cloud). They are primarily used when you need specific environment configurations, such as custom hardware (GPUs), specific operating systems, or access to internal resources behind a firewall.
Using self-hosted runners is a standard solution for accessing private internal network resources, as they can be placed directly within your on-premises or cloud-based private network.
Incorrect:
[Not B]
For GitHub Actions workflows with no specific environment configurations that must be triggered on GitHub-hosted infrastructure, you should typically use GitHub-hosted runners, not self-hosted ones.
Reference:
https://github.blog/enterprise-software/ci-cd/when-to-choose-github-hosted-runners-or-self-hosted-runners-with-github-actions

NEW QUESTION: 57
You need to perform a task that requires running a custom script in a GitHub Actions workflow.

The script must run within the workflow environment without requiring an external container or prebuilt image. What should you use?

A. a run step

B. a JavaScript action

C. a composite action

D. a Docker container action

Answer: A (LEAVE A REPLY)

To run a custom script directly in a GitHub Actions runner without an external container, you use the run step. You can write the script inline or execute a file already present in your repository.

Option 1: Inline Script

This is the simplest way to execute commands directly in the runner's shell.

```
jobs:
example-job:
runs-on: ubuntu-latest
steps:
- name: Run inline shell commands
run: |
echo "Starting custom script..."
python3 --version
echo "Hello from the runner environment!"
```

Option 2: Repository Script File

If your script is complex, store it in your repo (e.g., scripts/my-script.sh) and call it after using actions/checkout.

Reference:

https://github.com/orgs/community/discussions/47165

NEW QUESTION: 58

You installed specific software on a Linux self-hosted runner. You have users with workflows that need to be able to select the runner based on the identified custom software. Which steps should you perform to prepare the runner and your users to run these workflows? (Choose two.)

A. Create the group custom-software-on-linux and move the runner into the group.

B. Inform users to identify the runner based on the group.

C. Add the label custom-software to the runner.

D. Configure the webhook and network to enable GitHub to trigger workflow.

E. Add the label linux to the runner.

Answer: B,C (LEAVE A REPLY)

Once the runner is properly configured and labeled, users should be informed to select the specific runner by identifying the label or group name when defining the runner in their workflows.

Adding a custom label (like custom-software) to the runner makes it easier for users to select the runner in their workflows by using the runs-on key, which allows them to choose this specific runner based on its label.

NEW QUESTION: 59
As a developer, you are optimizing a GitHub workflow that uses and produces many different files. You need to determine when to use caching versus workflow artifacts. Which two statements are true? (Each correct answer presents part of the solution. Choose two.)
A. Use artifacts to access the GitHub Package Registry and download a package for a workflow.
B. Use caching when reusing files that change rarely between jobs or workflow runs.
C. Use caching to store cache entries for up to 30 days between accesses.
D. Use artifacts when referencing files produced by a job after a workflow has ended.
Answer: B,D (LEAVE A REPLY)
[B] Use caching when you want to reuse files that don't change often between jobs or workflow runs, such as build dependencies from a package management system.
[D] Use artifacts when you want to save files produced by a job to view after a workflow run has ended, such as built binaries or build logs.
Reference:
https://docs.github.com/en/enterprise-server@3.16/actions/tutorials/store-and-share-data

NEW QUESTION: 60
As a developer, you are using a Docker container action in your workflow. What is required for the action to run successfully?
A. The job env must be set to a Linux environment.
B. The action must be published to the GitHub Marketplace.
C. The referenced action must be hosted on Docker Hub.
D. The job runs-on must specify a Linux machine with Docker installed.
Answer: D (LEAVE A REPLY)
To run a Docker container action, the job's runs-on environment must be a Linux machine that has Docker installed.
Key Requirements
Linux OS Only: Docker container actions are only supported on Linux runners. They cannot run on Windows or macOS runners.
Docker Installed: The runner must have the Docker CLI and daemon installed and running to build and execute the action's container.
GitHub-Hosted Runners: If you use GitHub-hosted runners, you must specify an Ubuntu environment (e.g., runs-on: ubuntu-latest), as these come pre-configured with Docker.
Self-Hosted Runners: For self-hosted runners, you must manually ensure the machine is running Linux and has Docker installed.
Reference:

https://docs.github.com/actions/sharing-automations/creating-actions/creating-a-docker-container-action

NEW QUESTION: 61
As a developer, you have a 10-MB data set that is required in a specific workflow. Which steps should you perform so the dataset is stored encrypted and can be decrypted during the workflow? (Choose three.)
A. Encrypt the dataset.
B. Leverage the actions/download-secret action in the workflow.
C. Store the dataset in a GitHub encrypted secret.
D. Store the encryption keys in a GitHub encrypted secret.
E. Compress the dataset
F. Commit the encrypted dataset to the same repository as the workflow
G. Create a GitHub encrypted secret with the Large object option selected and upload the dataset.
Answer: A,C,D (LEAVE A REPLY)
First, the dataset should be encrypted before being stored. This ensures that the data is protected when stored in a repository.
The encrypted dataset can be stored in a GitHub secret, ensuring it is securely kept and not exposed publicly.
The encryption key needed to decrypt the dataset should also be stored in a GitHub secret to maintain security during the workflow, allowing access only when needed.

NEW QUESTION: 62
As a developer, you need to integrate a GitHub Actions workflow with a third-party code quality provider that uses the Checks API. How should you trigger a follow-up workflow?
A. Add the deployment webhook event as a trigger for the workflow when the code quality integration is completed
B. Add the pull_request webhook event as a trigger for the workflow when the code quality integration is synchronized
C. Add the check_run webhook event as a trigger for the workflow when the code quality integration is completed

D. Add the workflow_run webhook event as a trigger for the workflow for the code quality integration name

Answer: C (LEAVE A REPLY)

NEW QUESTION: 63

As a developer, you need to use GitHub Actions to deploy a microservice that requires runtime access to a secure token. This token is used by a variety of other microservices managed by different teams in different repos. To minimize management overhead and ensure the token is secure, which mechanisms should you use to store and access the token? (Choose two.)

A. Store the token in a configuration file in a private repository. Use GitHub Actions to deploy the configuration file to the runtime environment.

B. Store the token as a GitHub encrypted secret in the same repo as the code. Create a reusable custom GitHub Action to access the token by the microservice at runtime.

C. Use a corporate non-GitHub secret store (e.g., HashiCorp Vault) to store the token. During deployment, use GitHub Actions to store the secret in an environment variable that can be accessed at runtime.

D. Store the token as a GitHub encrypted secret in the same repo as the code. During deployment, use GitHub Actions to store the secret in an environment variable that can be accessed at runtime.

E. Store the token as an organizational-level encrypted secret in GitHub. During deployment, use GitHub Actions to store the secret in an environment variable that can be accessed at runtime.

Answer: C,E (LEAVE A REPLY)

Using a corporate secret store like HashiCorp Vault provides a secure, centralized location for sensitive information. GitHub Actions can then retrieve and store the token securely during deployment by setting it as an environment variable, ensuring the token remains secure and accessible at runtime.

Storing the token as an organizational-level encrypted secret in GitHub ensures it is accessible across multiple repositories, minimizing management overhead. GitHub Actions can then use this secret during deployment by setting it as an environment variable, allowing the microservice to access it securely at runtime.

NEW QUESTION: 64

You need to pass values from a runner to other parts of a GitHub Actions workflow.

Which two actions should you perform? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

A. Set a debug message.

B. Read from the environment variables.

C. Configure the output parameters.

D. Create environment variables by writing to the GITHUB_ENV file.

Answer: B,D (LEAVE A REPLY)

To pass a value between steps within the same job, you write to the $GITHUB_ENV file.

[D] 1. Write the value

Use a shell command to echo the variable name and value into the environment file:

- name: Set the value

run: echo "MY_VARIABLE=hello-world" >> $GITHUB_ENV

[B] 2. Read the value

In any subsequent step within that same job, you can access it like a standard environment variable or via the env context:

- name: Use the value

run: echo "The value is ${{ env.MY_VARIABLE }}"

Note: Environment variables set this way are only available to subsequent steps in the same job.

If you need to pass a value to a different job, you must use outputs instead.

Reference:

https://docs.github.com/actions/learn-github-actions/variables

NEW QUESTION: 65

As a developer, how can you identify a JavaScript action on GitHub?

A. The action's repository includes a js.yml file in the .github/workflows directory.

B. The action's repository name includes the keyword "JavaScript."

C. The action.yml metadata file has the runs.using value set to node16.

D. The action.yml metadata file references a package.json file.

Answer: D (LEAVE A REPLY)

Creating a JavaScript action, example

Prerequisites

Before you begin, you'll need to download Node.js and create a public GitHub repository.

* steps omitted *

From your terminal, initialize the directory with npm to generate a package.json file.

npm init -y

--

Adding actions toolkit packages

The actions toolkit is a collection of Node.js packages that allow you to quickly build JavaScript actions with more consistency.

When you commit and push your code, your updated repository should look like this:

hello-world-javascript-action/

├── action.yml

├── dist/

│ └── index.js

├── package.json

├── package-lock.json

├── README.md
├── rollup.config.js
└── src/
└── index.js
Reference:
https://docs.github.com/en/actions/tutorials/create-actions/create-a-javascript-action#creating-an- action-metadata-file

NEW QUESTION: 66
What will the output be for the following event trigger block in a workflow?

A. It throws a workflow syntax error, pointing to the types definition in issue_comment event.
B. It runs the workflow when an issue or issue comment in the workflow's repository is created or modified.
C. It runs the workflow when an issue is edited or when an issue comment created.
D. It runs the workflow when an issue is created or edited, or when an issue or pull request comment is created.
E. It throws a workflow syntax error, pointing to the types definition in issues event.
Answer: C (LEAVE A REPLY)
Based on your configuration, the workflow will trigger in these three specific scenarios:
Issue Opened: When a brand-new issue is created.
Issue Edited: When the title or body of an existing issue is changed.
Comment Created: When someone adds a new comment to an issue or pull request.
Reference:
https://forgejo.org/docs/next/user/actions/reference

NEW QUESTION: 67
As a DevOps engineer developing a JavaScript action, you need to include annotations to pass warning messages to workflow runners. Which code snippet can you use to implement an annotation in your Actions?
As a DevOps engineer developing a JavaScript action, you need to include annotations to pass warning messages to workflow runners. Which code snippet can you use to implement an annotation in your Actions?
A. core.info('Something went wrong, but it\'s not bad enough to fail the build.')
B. core.notice('Something went wrong, but it\'s not bad enough to fail the build.')

C. core.warning('Something went wrong, but it\'s not bad enough to fail the build.')
D. core.warn('Something went wrong, but it\'s not bad enough to fail the build.')
Answer: C (LEAVE A REPLY)
The core.warning() function from the @actions/core package is used to create a warning message in the workflow logs. This is an annotation type that informs users about issues that don't require failing the build but still need attention.

NEW QUESTION: 68
Which default GitHub environment variable indicates the name of the person or app that initiated a workflow?
A. GITHUB_ACTOR
B. GITHUB USER
C. ENV_ACTOR
D. GITHUB_WORKFLOW_ACTOR
Answer: A (LEAVE A REPLY)
GITHUB_ACTOR : The name of the person or app that initiated the workflow.
Reference:
https://graphite.dev/guides/github-actions-variables
GITHUB_ACTOR : The name of the person or app that initiated the workflow.

NEW QUESTION: 69
You are a DevOps engineer working on deployment workflows. You need to execute the deploy job only if the current branch name is feature-branch. Which code snippet will help you to implement the conditional execution of the job?
A. jobs:
deploy:
if: github.ref_name == ' feature-branch '
runs-on: ubuntu-latest
steps:
- uses: actions/checkout@v3
B. jobs:
deploy:
if: github.ref.name == ' feature-branch '
runs-on: ubuntu-latest
steps:
- uses: actions/checkout@v3
C. jobs:
deploy:
if: github.branch_name == ' feature-branch '
runs-on: ubuntu-latest
steps:

- uses: actions/checkout@v3
D. jobs:
deploy:
if: github.branch.name == ' feature-branch '
runs-on: ubuntu-latest
steps:
- uses: actions/checkout@v3
Answer: A (LEAVE A REPLY)
The correct GitHub Actions context property for the short branch or tag name that triggered the workflow is github.ref_name. A job-level if: condition can use this context to control whether the job runs. Therefore, if:
github.ref_name == ' feature-branch ' correctly limits the deploy job to the feature-branch branch. The other options use invalid context paths such as github.ref.name, github.branch_name, and github.branch.name; these are not recognized GitHub Actions properties. In exam terms, this question validates knowledge of workflow expressions, GitHub context variables, and conditional job execution. The important distinction is that github.ref contains the full ref path, while github.ref_name provides the branch or tag name directly.

NEW QUESTION: 70
Disabling a workflow allows you to stop a workflow from being triggered without having to delete the file from the repo. In which scenarios would temporarily disabling a workflow be most useful? (Choose two.)
A. A workflow sends requests to a service that is down.
B. A workflow error produces too many, or wrong, requests, impacting external services negatively.
C. A workflow is configured to run on self-hosted runners
D. A workflow needs to be changed from running on a schedule to a manual trigger
E. A runner needs to have diagnostic logging enabled.
Answer: A,B (LEAVE A REPLY)
If a workflow depends on an external service that is down, disabling the workflow temporarily will prevent it from running and sending requests to the service, thus avoiding failed requests or unnecessary retries.
If a workflow is causing a negative impact on external services by generating too many requests or incorrect data due to a bug, temporarily disabling the workflow will stop this behavior while the issue is fixed.

NEW QUESTION: 71
What menu options in a repository do you need to select in order to use a starter workflow that is provided by your organization?
A. Actions > Load workflow

B. Workflow > New workflow
C. Workflow > Load workflow
D. Actions > New workflow
Answer: D (LEAVE A REPLY)
To use a starter workflow provided by your organization, you need to go to the Actions tab in the repository and select New workflow. This option allows you to either create a new workflow or select from a list of available workflow templates, including starter workflows provided by your organization.

NEW QUESTION: 72
By default, which workflows can use an action stored in internal repository? (Each answer presents a complete solution. Choose two.)
A. selected public repositories outside of the enterprise
B. internal repositories owned by the same organization as the enterprise
C. public repositories owned by the same organization as the enterprise
D. private repositories owned by an organization of the enterprise
Answer: B,D (LEAVE A REPLY)
Any actions or reusable workflows stored in the internal or private repository can be used in workflows defined in other internal or private repositories owned by the same organization, or by any organization owned by the enterprise.
Reference:
https://docs.github.com/actions/creating-actions/sharing-actions-and-workflows-with-your-enterprise

NEW QUESTION: 73
In which of the following scenarios should you use self-hosted runners? Each correct answer presents a complete solution.
NOTE: Each correct answer is worth one point.
A. when the workflow jobs must be run on Windows 10
B. when jobs must run for longer than 6 hours
C. when you want to use macOS runners
D. when GitHub Actions minutes must be used for the workflow runs
E. when a workflow job needs to install software from the local network
Answer: B,E (LEAVE A REPLY)
Self-hosted runners are appropriate when GitHub-hosted runner limits or environment restrictions do not meet the workload requirement. GitHub-hosted runner jobs have a 6-hour job execution limit, while self-hosted runner jobs can run for up to 5 days, so jobs that must run longer than 6 hours require self-hosted runners. Self-hosted runners also let an organization control the machine, operating system, software, hardware, and network location. Therefore, if a workflow job needs to install or access software from the local network, a self-hosted runner is the correct choice. macOS and Windows runners are

available as GitHub-hosted options, and GitHub Actions minutes apply to GitHub-hosted usage, not self-hosted runner execution.

NEW QUESTION: 74
What are the two ways to pass data between jobs? (Choose two.)
A. Use the copy action with restore parameter to restore the data from the cache
B. Use the copy action to save the data that should be passed in the artifacts folder.
C. Use the copy action with cache parameter to cache the data
D. Use data storage.
E. Use job outputs
F. Use artifact storage.
Answer: E,F (LEAVE A REPLY)
Job outputs are used to pass data from one job to another in a workflow. A job can produce output values (like variables or files), which can be referenced by subsequent jobs using the needs keyword and ${{ steps.step_id.outputs.output_name }} syntax.
Artifact storage allows data (such as files or results) to be saved by a job and then retrieved by another job in a later step. This is commonly used for passing large amounts of data or files between jobs.

NEW QUESTION: 75
Your organization needs to simplify reusing and maintaining automation in your GitHub Enterprise Cloud.
Which components can be directly reused across all repositories in an organization? (Choose three.)
A. self-hosted runners
B. actions stored m private repositories in the organization
C. encrypted secrets
D. custom Docker actions stored in GitHub Container Registry
E. actions stored in an organizational partition in the GitHub Marketplace
F. workflow templates
Answer: B,C,F (LEAVE A REPLY)
Actions stored in private repositories within the organization can be reused across all repositories by referencing them in workflows. This ensures a centralized way of maintaining custom actions.
Encrypted secrets can be accessed across repositories in the same organization, making it easy to store sensitive data (like API keys or tokens) securely while allowing multiple workflows to access them.
Workflow templates allow you to create reusable templates for workflows that can be shared across repositories within the organization. This makes it easier to standardize processes and automate them across multiple projects.

NEW QUESTION: 76
Which of the following is the lowest repository permission you need to have for downloading workflow artifacts?
A. Triage
B. Admin
C. Maintain
D. Write
E. Read
Answer: E (LEAVE A REPLY)
Downloading workflow artifacts
You can download archived artifacts before they automatically expire.
Who can use this feature?
People who are signed into GitHub and have read access to a repository can download workflow artifacts.
Reference:
https://docs.github.com/en/actions/how-tos/manage-workflow-runs/download-workflow-artifacts

NEW QUESTION: 77
What are the advantages of using a matrix strategy in a job definition? (Choose two.)
A. It can test code in multiple versions of a programming language.
B. It can decrease the costs for running multiple combinations of programming language/operating systems.
C. It can run up to 512 jobs per workflow run.
D. It can test code in multiple operating systems.
Answer: A,D (LEAVE A REPLY)
A matrix strategy allows you to define different versions of a programming language (or any other environment setting) and run tests on each version simultaneously. This is particularly useful for testing code compatibility across different versions of a language.
A matrix strategy can also be used to test code on multiple operating systems (e.g., Windows, macOS, Linux) by defining these operating systems as matrix variables. This enables cross-platform testing within the same workflow.

NEW QUESTION: 78
You have exactly one Windows x64 self-hosted runner, and it is configured with custom tools.
Which syntax could you use in the workflow to target that runner?
A. runs-on: [self-hosted, windows, x64]
B. self-hosted: [windows-x64]
C. runs-on: windows-latest
D. self-hosted: [windows, x64]
Answer: A (LEAVE A REPLY)
Using self-hosted runners in a workflow
To use self-hosted runners in a workflow, you can use labels or groups to specify the runner for a job.
Using default labels to route jobs
A self-hosted runner automatically receives certain labels when it is added to GitHub Actions.
These are used to indicate its operating system and hardware platform:
self-hosted: Default label applied to self-hosted runners.
linux, windows, or macOS: Applied depending on operating system.
x64, ARM, or ARM64: Applied depending on hardware architecture.
You can use your workflow's YAML to send jobs to a combination of these labels. In this example, a self-hosted runner that matches all three labels will be eligible to run the job:
runs-on: [self-hosted, linux, ARM64]
self-hosted - Run this job on a self-hosted runner.
linux - Only use a Linux-based runner.
ARM64 - Only use a runner based on ARM64 hardware.
Reference:
https://docs.github.com/en/actions/how-tos/manage-runners/self-hosted-runners/use-in-a-workflow

NEW QUESTION: 79
What are the two ways to pass data between jobs? (Each correct answer presents part of the solution. Choose two.)
A. Use data storage.
B. Use the copy action with cache parameter to cache the data.
C. Use the copy action with restore parameter to restore the data from the cache.
D. Use artifact storage.
E. Use job outputs.
F. Use the copy action to save the data that should be passed in the artifacts folder.
Answer: D,E (LEAVE A REPLY)
Passing information between jobs

[D] Store and share data with workflow artifacts
Use artifacts to share data between jobs in a workflow and store data once that workflow has completed.
[E] You can define outputs to pass information from one job to another.
Reference:
https://docs.github.com/en/actions/tutorials/store-and-share-data
https://docs.github.com/en/actions/how-tos/write-workflows/choose-what-workflows-do/pass-job- outputs

NEW QUESTION: 80
Which files are required for a Docker container action in addition to the source code? (Choose two.)
A. Dockerfile
B. Actionfile
C. metadata.yml
D. action.yml
Answer: A,D (LEAVE A REPLY)
Dockerfile: The Dockerfile is required for Docker container actions. It defines the environment for the action, specifying the base image, dependencies, and any commands to set up the action's runtime inside the container.
action.yml: The action.yml file is required for all GitHub Actions, including Docker container actions. It contains metadata about the action, including the inputs, outputs, and the runtime environment (which in this case is Docker, defined under runs.using).

NEW QUESTION: 81
As a developer, you need to use GitHub Actions to deploy a microservice that requires runtime access to a secure token. This token is used by a variety of other microservices managed by different teams in different repos. To minimize management overhead and ensure the token is secure, which mechanisms should you use to store and access the token? (Each correct answer presents a complete solution. Choose two.)
A. Store the token as a GitHub encrypted secret in the same repo as the code. During deployment, use GitHub Actions to store the secret in an environment variable that can be accessed at runtime.
B. Use a corporate non-GitHub secret store (e.g., HashiCorp Vault) to store the token. During deployment, use GitHub Actions to store the secret in an environment variable that can be accessed at runtime.
C. Store the token as an organizational-level encrypted secret in GitHub. During deployment, use GitHub Actions to store the secret in an environment variable that can be accessed at runtime.
D. Store the token as a GitHub encrypted secret in the same repo as the code. Create a reusable custom GitHub Action to access the token by the microservice at runtime.

E. Store the token in a configuration file in a private repository. Use GitHub Actions to deploy the configuration file to the runtime environment.

Answer: B,C (LEAVE A REPLY)

[B] Using a corporate secret store like HashiCorp Vault provides a secure, centralized location for sensitive information. GitHub Actions can then retrieve and store the token securely during deployment by setting it as an environment variable, ensuring the token remains secure and accessible at runtime.

[C] Storing the token as an organizational-level encrypted secret in GitHub ensures it is accessible across multiple repositories, minimizing management overhead. GitHub Actions can then use this secret during deployment by setting it as an environment variable, allowing the microservice to access it securely at runtime.

NEW QUESTION: 82

What is the most secure way to store sensitive information in GitHub Actions workflows?

A. Use environment variables in the workflow and store sensitive data directly in the variables.

B. From the GitHub repository settings, store sensitive data as unencrypted text.

C. Store the sensitive information as plain text in the workflow YAML file.

D. Use GitHub Enterprise Managed User (OIDC) integration to dynamically fetch secrets from a third- party secrets manager.

Answer: D (LEAVE A REPLY)

The most secure option listed is to use OIDC-based integration to obtain short-lived credentials from an external secrets manager or cloud provider. This avoids storing long-lived sensitive values directly in workflow YAML, repository text, or regular environment variables. Options A, B, and C are insecure because they expose or persist sensitive information in places that can be read, logged, committed, or mismanaged. With OIDC, the workflow requests a token from GitHub's OIDC provider and exchanges it with the trusted external provider for a short-lived credential. This reduces secret sprawl and supports stronger authorization controls. GitHub documents OIDC as a way to avoid duplicating long- lived cloud credentials as GitHub secrets and to use short-lived access tokens instead.

NEW QUESTION: 83

Which scopes are available to define custom environment variables within a workflow file? (Choose three.)

A. the entire workflow, by using env at the top level of the workflow file

B. all jobs being run on a single Actions runner, by using runner.env at the top of the workflow file

C. the entire stage, by using env at the top of the defined build stage

D. within the run attribute of a job step

E. the contents of a job within a workflow, by using jobs.env

F. a specific step within a job, by using jobs..steps[*].env

Answer: A,E,F (LEAVE A REPLY)

Defining environment variables for a single workflow

To set a custom environment variable for a single workflow, you can define it using the env key in the workflow file. The scope of a custom variable set by this method is limited to the element in which it is defined. You can define variables that are scoped for:

[A] The entire workflow, by using env at the top level of the workflow file.

[E] The contents of a job within a workflow, by using jobs.<job_id>.env.

[F] A specific step within a job, by using jobs.<job_id>.steps[*].env.

Reference:

https://docs.github.com/en/actions/how-tos/write-workflows/choose-what-workflows-do/use- variables

NEW QUESTION: 84

Which of the following statements are true regarding the use of GitHub Actions on a GitHub Enterprise Server instance? (Each correct answer presents a complete solution. Choose three.)

A. Actions created by GitHub are automatically available and cannot be disabled.

B. Third-party actions can be used on GitHub Enterprise Server by configuring GitHub Connect.

C. Most GitHub-authored actions are automatically bundled for use on GitHub Enterprise Server.

D. Use of GitHub Actions on GitHub Enterprise Server requires a persistent internet connection.

E. Actions must be defined in the .github repository.

F. Third-party actions can be manually synchronized for use on GitHub Enterprise Server.

Answer: B,C,F (LEAVE A REPLY)

[B] If you want your GitHub Enterprise Server instance to use third-party custom actions, you need to enable GitHub Connect.

[C] Most official GitHub-authored actions are automatically bundled with GitHub Enterprise Server, and are captured at a point in time from GitHub Marketplace.

[F] Third-party actions can be manually synchronized to GitHub Enterprise Server (GHES) to be used in workflows without requiring an internet connection to GitHub.com. This is done using the actions-sync tool to download actions from GitHub.com and transfer them to the GHES instance.

This manual method provides an alternative to using GitHub Connect, offering stricter control over which actions are available on the GHES instance.

Reference:

https://docs.github.com/en/enterprise-server@3.17/get-started/exploring-integrations/about- using-integrations

https://docs.github.com/en/enterprise-server@3.17/admin/managing-github-actions-for-your- enterprise/managing-access-to-actions-from-githubcom/about-using-actions-in-your-enterprise

NEW QUESTION: 85
What is the proper syntax to reference the system-provided run number variable?
A. ${{var.GITHUB_RUN_NUMBER}}
B. ${{env.GITHUB_RUN_NUMBER}}
C. $GITHUB_RUN_NUMBER
D. ${{GITHUB_RUN_NUMBER}}
E. $github.run_number
Answer: D (LEAVE A REPLY)
Default environment variables
The default environment variables that GitHub sets are available to every step in a workflow.
Because default environment variables are set by GitHub and not defined in a workflow, they are not accessible through the env context [Not B]. However, most of the default variables have a corresponding, and similarly named, context property. For example, the value of the GITHUB_REF variable can be read during workflow processing using the ${{ github.ref }} context property.
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/variables

NEW QUESTION: 86
Which of the following scenarios would require the use of self-hosted runners instead of GitHub- hosted runners?
A. running more than the three concurrent workflows supported by GitHub-hosted runners
B. performing builds on macOS
C. exceeding 50,000 monthly minutes of build time
D. using specialized hardware configurations required for workflows
E. using Docker containers as part of the workflow
Answer: (SHOW ANSWER)
The use of self-hosted runners is required when your workflows depend on specialized hardware configurations that are not available through standard GitHub-hosted runners. While GitHub offers "larger runners" with GPU support for certain paid plans, self-hosted runners provide the most flexibility for highly specific or proprietary hardware needs.
Scenarios Requiring Specialized Hardware
Specific GPU Models: Workflows for AI/ML training or intensive graphics rendering that require specific NVIDIA or other specialized GPU architectures.

Alternative CPU Architectures: When you must build or test on specific ARM processors, legacy x86 32-bit systems, or other non-standard architectures not supported by GitHub's managed pool.
High-Resource Requirements: Tasks needing massive amounts of RAM (beyond 256 GB) or high-core counts for extreme parallel processing.
Custom Peripherals: Workflows that need physical access to hardware connected via USB, PCIe, or other local interfaces (e.g., embedded systems testing or hardware-in-the-loop).
Reference:
https://docs.github.com/en/actions/how-tos/manage-runners/self-hosted-runners/use-in-a-workflow

NEW QUESTION: 87
As a DevOps engineer, you are trying to leverage an organization secret in a repo. The value received in the workflow is not the same as that set in the secret. What is the most likely reason for the difference?
A. There is a different value specified at the rego level.
B. There is a different value specified at the workflow level.
C. The Codespace secret doesn ' t match the expected value.
D. The Encrypt Secret setting was not configured for the secret.
E. There is a different value specified at the enterprise level.
Answer: A (LEAVE A REPLY)
GitHub secrets are defined at different levels: organization, repository, and sometimes at the workflow level.
If a secret is defined at both the organization level and the repository level, the repository-level secret will take precedence. So, if the value of the secret differs between these levels, the workflow will use the value from the repository level instead of the organization level.

NEW QUESTION: 88
Which two actions ensure that a GitHub Actions workflow can be triggered manually? Each correct answer presents a complete solution.
NOTE: Each correct selection is worth one point.
A. Define a workflow_dispatch event and specify inputs that enable users to provide values when triggering the workflow manually.
B. Use a schedule event and add a manual: false parameter.
C. Define a workflow_dispatch event in the on field of the workflow YAML file.
D. Trigger a workflow manually by using a workflow_call event that has a button on the UI.
Answer: (SHOW ANSWER)
The correct manual trigger for a GitHub Actions workflow is workflow_dispatch. Defining workflow_dispatch under the on key enables the workflow to be started manually from the GitHub UI, GitHub CLI, or API. Inputs are optional but valid; they allow the user to provide

values when manually starting the workflow, so option A is also correct. A schedule trigger is for cron-based automatic execution, not manual execution. workflow_call is used to make a workflow reusable by another workflow; it does not create the manual "Run workflow" button. Therefore, A and C are the correct selections. GitHub's documentation confirms that manual workflows require workflow_dispatch and can include custom inputs.

NEW QUESTION: 89
GitHub-hosted runners support which capabilities? (Each correct answer presents a complete solution. Choose two.)
A. support for Linux, Windows, and macOS
B. support for a variety of Linux variations including CentOS, Fedora, and Debian
C. requiring a payment mechanism (e.g., credit card) to use for private repositories
D. automatic file-system caching between workflow runs
E. automatic patching of both the runner and the underlying OS
Answer: A,E (LEAVE A REPLY)
[A, not B] Each GitHub-hosted runner is a new virtual machine (VM) hosted by GitHub with the runner application and other tools preinstalled, and is available with Ubuntu Linux, Windows, or macOS operating systems.
[E] When you use a GitHub-hosted runner, machine maintenance and upgrades are taken care of for you.
Reference:
https://docs.github.com/en/enterprise-server@3.17/actions/concepts/runners/github-hosted- runners

NEW QUESTION: 90
You need to create a reusable GitHub Actions workflow template named ci.yml. The solution must ensure that ci.yml appears on the New workflow interface of GitHub Actions. Where should you store ci.yml?
A. .github/workflow-templates
B. .github/templates
C. the root directory of each repository
D. .github/workflows
Answer: A (LEAVE A REPLY)
To ensure your reusable GitHub Actions workflow template appears in the New workflow interface (also known as the starter workflow picker), it must be stored in a specific directory within a special repository in your organization.
Required Storage Location
Repository Name: .github (This must be a public repository at the root of your organization).
Directory Path: workflow-templates.
Note:

Essential Files to Include
For the template to be correctly indexed and displayed in the interface, you must include two files in the workflow-templates directory:
The Workflow File: A standard YAML file (e.g., ci-template.yml) containing the workflow definition.
The Metadata File: A JSON file that must have the same name as the workflow file but with a .properties.json extension (e.g., ci-template.properties.json).
Reference:
https://docs.github.com/en/enterprise-server@3.19/actions/using-workflows/creating-starter- workflows-for-your-organization