ATMIC NETWORKS
YOUR GUIDE TO GLOBAL CERTIFIATION

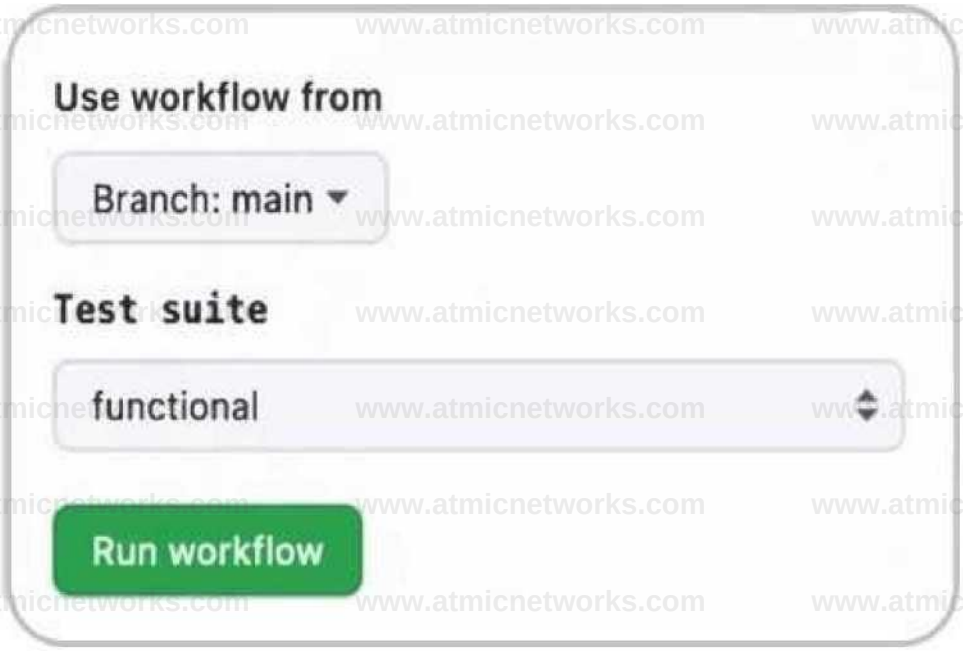
"Please note that these files may not be up to date. However, the questions will help you understand the exam format and typical question patterns."

www.atmicnetworks .com

Warning: Keep connected with our support team for latest updates

Question: 1

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

A. Option A B. Option B C. Option C D. Option D

Answer: C

Explanation:

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.

Question: 2

How many jobs will result from the following matrix configuration?

```
strategy:
  matrix:
    color: [green, pink]
    animal: [owl, magpie]
    include:
      - color: blue
        animal: owl
      - color: pink
        animal: magpie
```

A. 3 jobs
B. 4 jobs
C. 5 jobs
D. 6 jobs

Answer: D

Explanation:

The matrix configuration specifies two variables: color and animal. The color variable has 2 values (green and pink), and the animal variable has 2 values (owl and magpie). This would result in 4 combinations (2 color values × 2 animal values). Additionally, the include section introduces two more combinations (color: blue and animal: owl; color: pink and animal: magpie).

Question: 3

As a developer, which workflow steps should you perform to publish an image to the GitHub Container Registry? (Choose three.)

A. Use the actions/setup-docker action

B. Authenticate to the GitHub Container Registry.
C. Build the container image.
D. Push the image to the GitHub Container Registry
E. Pull the image from the GitHub Container Registry.

Answer: A, B, D

Explanation:
A . Use the actions/setup-docker action
B . Authenticate to the GitHub Container Registry.
C . Build the container image.
D . Push the image to the GitHub Container Registry
E . Pull the image from the GitHub Container Registry.

Question: 4

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

Explanation:

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.

Question: 5

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

Explanation:

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.

Question: 6

Which of the following commands will set the $FOO environment variable within a script, so that it may be used in subsequent workflow job steps?

A. run: echo "::set-env name=FOO::bar"

B. run: echo "FOO=bar" >> $GITHUB_ENV

C. run: echo ${{ $FOO=bar }}

D. run: export FOO=bar

Answer: B

Explanation:

The $GITHUB_ENV environment variable is used to set environment variables that persist across steps in a workflow job. By echoing FOO=bar into $GITHUB_ENV, the variable FOO will be available in subsequent steps within the same job.

Question: 7

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?

A. via the .github repository owned by the organization

B. via repositories owned by the organization
C. via the GitHub Marketplace
D. via a repository owned by a third party

Answer: B

Explanation:

To prevent reaching the storage limit for GitHub artifacts and packages, you should manage and clean up artifacts and packages stored in repositories owned by your organization. This includes deleting unnecessary artifacts and managing the lifecycle of packages, as they contribute directly to your organization's storage quota.

Question: 8

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®vl 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®vl with:
          node-version: 12
      - run: npm ci
      - uses: JS-DevTools/npm-publish@vl 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

Explanation:

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.

Question: 9

In which scenarios could the GITHUB_TOKEN be used? (Choose two.)

A. to leverage a self-hosted runner
B. to create a repository secret
C. to publish to GitHub Packages
D. to create issues in the repo
E. to read from the file system on the runner
F. to add a member to an organization

Answer: C, D

Explanation:

The GITHUB_TOKEN is automatically provided by GitHub in workflows and can be used to authenticate API requests to GitHub, including publishing packages to GitHub Packages.
The GITHUB_TOKEN is also used to authenticate API requests for actions like creating issues, commenting, or interacting with pull requests within the same repository.

Question: 10

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

Explanation:

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.

Question: 11

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

A)

```
outputs:
  too:
    description: Some value
```

B)

```
outputs:
  - key: too description: Some value
```

C)

```
outputs:
  too: value: Some value
```

D)

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

A. Option A

B. Option B

C. Option C

D. Option D

Answer: C

Explanation:

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.

Question: 12

When reviewing an action for use, what file defines its available inputs and outputs?

A. inputs.yml
B. config.json
C. defaults.json
D. workflow.yml
E. action.yml

Answer: E

Explanation:

The action.yml file defines the inputs and outputs for a GitHub Action. This file contains metadata about the action, including the required inputs and outputs, as well as other configurations like the action's description, runs, and environment setup.

Question: 13

How should you install the bats NPM package in your workflow?

A)

```
jobs:
  example-j ob:
    steps:
      - npm install -g bats
```

B)

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

C)

```
jobs:
  runs-on: ubuntu-latest
```

```
  - run: npm install -g bats
```

D)

```
jobs:
  example-job:
    runs-on: ubuntu-latest
    steps:
      - run: npm install -g bats
```

A. Option A B. Option B C. Option C D. Option D

Answer: D

Explanation:

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.

Question: 14

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

Explanation:

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.

Question: 15

As a DevOps engineer, you are developing a container action. You need to execute a cleanup script after completing the main script execution. Which code block should be used to define the cleanup script?

A.

```
runs:
  using: 'docker'
```

```
   image: 'Dockerfile'
   entrypoint: 'main.sh'
   post: 'cleanup.sh'
```

B.

```
runs:
   using: 'docker'
   image: 'Dockerfile' entrypoint: 'main.sh' post-if:
   'cleanup.sh'
```

C.

```
runs:
   using: 'docker'
   image: 'Dockerfile' entrypoint: 'main.sh' after:
   'cleanup.sh'
```

D.

```
runs:
   using: 'docker'
   image: 'Dockerfile' entrypoint: 'main.sh' post-
   entrypoint: 'cleanup.sh'
```

A. Option A B. Option B C. Option C D. Option D

Answer: A

Explanation:

The correct syntax for specifying a cleanup script to be run after the main entry point is executed in a Docker-based GitHub Action is to use the post key. This ensures that cleanup.sh runs after the main script (main.sh) has completed.

Question: 16

Which default GitHub environment variable indicates the name of the person or app that initiated a workflow?

A. ENV_ACTOR
B. GITHUB_WORKFLOW_ACTOR
C. GITHUB_ACTOR
D. GITHUB_USER

Answer: C

Explanation:

The GITHUB_ACTOR environment variable indicates the name of the person or app that initiated the workflow.

This variable is automatically provided by GitHub in the workflow and can be used to identify the user or application triggering the workflow.

Question: 17

You are a developer, and your container jobs are failing on a self-hosted runner. Which requirements must you check to ensure that the self-hosted runner is properly configured? (Choose two.)

A. The self-hosted runner is running a Linux operating system.
B. The self-hosted runner is running a Windows operating system.
C. Docker is installed on the self-hosted runner.
D. Kubernetes is installed on the self-hosted runner.
E. The service status of Kubernetes is "active".

Answer: A, C

Explanation:

While Docker can run on various operating systems, Linux is often the most common and preferred OS for containerized workloads. Docker works well on Linux and is a widely-used platform for running containers. For container jobs to run on a self-hosted runner, Docker must be installed and properly configured on the runner. Docker is required to build and run containerized workloads in a GitHub Actions workflow.

Question: 18

Which choices represent best practices for publishing actions so that they can be consumed reliably? (Choose two.)

A. repo name
B. tag
C. commit SHA
D. organization name
E. default branch

Answer: B, C

Explanation:

Using a tag is a best practice because tags are immutable and represent a fixed version of your action. By referencing tags, consumers of your action can be assured they are using a stable and specific version of the action, which helps in avoiding issues with breaking changes.
The commit SHA is another reliable way to specify a particular version of an action. By referencing a specific commit SHA, consumers can ensure they are using exactly the code that was written at that moment, avoiding the potential for changes in the future.

Question: 19

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

Explanation:

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.

Question: 20

GitHub-hosted runners support which capabilities? (Choose two.)

A. automatic patching of both the runner and the underlying OS
B. automatic file-system caching between workflow runs
C. support for Linux, Windows, and mac
D. support for a variety of Linux variations including CentOS, Fedora, and Debian
E. requiring a payment mechanism (e.g., credit card) to use for private repositories

Answer: C, D

Explanation:

GitHub-hosted runners automatically handle patching, meaning they will be kept up to date with the latest security updates and software patches for both the runner environment and the underlying operating system.
GitHub-hosted runners support Linux, Windows, and macOS, giving you flexibility to run workflows on different operating systems without needing to manage your own self-hosted runners.

Question: 21

You need to make a script to retrieve workflow run logs via the API. Which is the correct API to download a workflow run log?

A. POST /repos/:owner/:repo/actions/runs/:run_id
B. GET /repos/:owner/:repo/actions/artifacts/logs
C. GET /repos/:owner/:repo/actions/runs/:run_id/logs

D. POST /repos/:owner/:repo/actions/runs/:run_id/logs

Answer: C

Explanation:

The GET /repos/:owner/:repo/actions/runs/:run_id/logs API endpoint is used to retrieve the logs of a specific workflow run identified by run_id. This is the correct method for downloading logs from a workflow run.

Question: 22

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? (Choose two.)

A. Use caching when reusing files that change rarely between jobs or workflow runs.
B. Use artifacts when referencing files produced by a job after a workflow has ended.
C. Use caching to store cache entries for up to 30 days between accesses.
D. Use artifacts to access the GitHub Package Registry and download a package for a workflow

Answer: A, B

Explanation:

Caching is ideal for files that change rarely, such as dependencies or build outputs, as it speeds up subsequent workflow runs by reusing previously cached files instead of re-downloading or rebuilding them.

Artifacts are used for persisting files produced during a job that need to be used in later jobs or after the workflow has ended, allowing them to be downloaded or referenced later.

Question: 23

As a developer, you are designing a workflow and need to communicate with the runner machine to set environment variables, output values used by other actions, add debug messages to the output logs, and other tasks. Which of the following options should you use?

A. environment variables
B. workflow commands
C. self-hosted runners
D. enable debug logging
E composite run step

Answer: B

Explanation:

Workflow commands are special commands that allow you to interact with the runner, set environment variables, output values, add debug messages, and perform other tasks within the workflow. These commands

are used to modify the environment or influence the behavior of the GitHub Actions runner.

Question: 24

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: D

Explanation:

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.

Question: 25

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

Explanation:

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.

Question: 26

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

Explanation:

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.

Question: 27

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

Explanation:

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.

Question: 28

In which locations can actions be referenced by workflows? (Choose three.)

A. a separate public repository
B. an .action extension file in the repository
C. the same repository as the workflow
D. a published Docker container image on Docker Hub
E. the runs-on: keyword of a workflow file
F. the repository's Secrets settings page
G. a public NPM registry

Answer: A, C, D

Explanation:

Actions can be stored in a separate public repository and referenced in workflows by specifying the repository and action name.
Actions can also be stored in the same repository as the workflow and referenced directly by their path (e.g., ./.github/actions/my-action).
Actions can be packaged as Docker container images and published to Docker Hub. These can then be referenced in workflows by specifying the Docker image.

Question: 29

As a developer, what options should you recommend to implement standards for automation reuse? (Choose two.)

A. Create workflow templates and store them in the organization's .github repository.
B. Create reusable actions and workflows that can be called from other workflows.
C. Create a marketplace partition to publish reusable automation for the company.
D. Store shared corporate actions in subfolders in a defined and documented internally accessible repository.

Answer: A, B

Explanation:

Creating workflow templates in the organization’s .github repository allows the organization to standardize workflows and make them easily reusable across multiple repositories. This ensures consistency and simplifies maintenance.
Creating reusable actions and workflows that can be called from other workflows helps modularize and standardize automation tasks. These reusable components can be maintained centrally and called from different workflows across repositories.

Question: 30

As a developer, you need to make sure that only actions from trusted sources are available for use in your

GitHub Enterprise Cloud organization. Which of the following statements are true? (Choose three.)

A. Specific actions can individually be enabled for the organization, including version information.
B. GitHub-verified actions can be collectively enabled for use in the enterprise.
C. Actions can be restricted to only those available in the enterprise.
D. Actions created by GitHub are automatically enabled and cannot be disabled.
E. Individual third-party actions enabled with a specific tag will prevent updated versions of the action from introducing vulnerabilities.
F. Actions can be published to an internal marketplace.

Answer: A, B, F

Explanation:

You can enable specific actions for the organization by identifying them and providing version control, ensuring only trusted versions are used in workflows.
GitHub-verified actions can be enabled at the enterprise level, providing an extra layer of security by ensuring that only trusted actions are available to workflows.
Actions can be published to an internal marketplace, allowing organizations to share reusable actions securely within their enterprise without exposing them to the public.

Question: 31

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

Explanation:

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.

Question: 32

Where should workflow files be stored to be triggered by events in a repository?

A. .github/workflows/
B. .github/actions/
C. Nowhere; they must be attached to an act on in the GitHub user interface
D. anywhere

E. .workflows/

Answer: A

Explanation:

Workflow files must be stored in the .github/workflows/ directory of the repository. This is the standard location for GitHub Actions workflow files, and workflows in this directory are automatically triggered by events defined in the file, such as pushes, pull requests, or other GitHub events.

Question: 33

A development team has been using a Powershell script to compile and package their solution using existing tools on a Linux VM, which has been configured as a self-hosted runner. They would like to use the script as-is in an automated workflow. Which of the following should they do to invoke their script within a workflow step?

A. Configure a self-hosted runner on Windows with the requested tools.
B. Use the YAML powershell: step.
C. Run the pwsh2bash command to convert the script so it can be run on Linux.
D. Use the YAML shell: pwsh in a run step.
E. Use the actions/run-powershell action to invoke the script.

Answer: D

Explanation:

Since the self-hosted runner is configured on a Linux VM and the script is written in PowerShell, you can invoke the script using the pwsh (PowerShell Core) shell in a run step in the workflow. This ensures that the script runs as-is on the Linux runner, as PowerShell Core (pwsh) is cross-platform and supports Linux.

Question: 34

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

Explanation:

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.

Question: 35

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: B

Explanation:

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.

Question: 36

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. exceeding 50,000 monthly minutes of build time
C. using Docker containers as part of the workflow
D. using specialized hardware configurations required for workflows
E. performing builds on macOS

Answer: A, D

Explanation:

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.

Question: 37

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

Explanation:

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.

Question: 38

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

Explanation:

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.

Question: 39

As a developer, what is the safest way to reference an action to prevent modification of the underlying code?

A. Use a commit hash.
B. Use a branch name.
C. Use a patch release tag.
D. Use a major release tag.

Answer: A

Explanation:

Using a commit hash is the safest method because it references a specific point in time in the repository's history. This ensures that the action is locked to that exact version and will not be affected by any future changes or modifications to the codebase. Even if the action is updated later, your workflow will continue using the specific commit you referenced.

Question: 40

Without the need to use additional infrastructure, what is the simplest and most maintainable method for configuring a workflow job to provide access to an empty PostgreSQL database?

A. Use service containers with a Postgres database from Docker hub.
B. Run the actions/postgres action in a parallel job.
C. It is currently impossible to access the database with GitHub Actions.
D. Dynamically provision and deprovision an environment.

Answer: A

Explanation:

GitHub Actions supports the use of service containers, which allows you to spin up a PostgreSQL database (or any other service) in a Docker container during your workflow. You can pull a PostgreSQL image from Docker Hub, and the container will automatically be available to your workflow job. This method requires no additional infrastructure and is easy to configure and maintain, as you simply define the container in the workflow file.

Question: 41

Which of the following scenarios requires a developer to explicitly use the GITHUB_TOKEN or github.token secret within a workflow? (Choose two.)

A. passing the GITHUB_TOKEN secret to an action that requires a token as an input
B. making an authenticated GitHub API request
C. checking out source code with the actions/checkout@v3 action
D. assigning non-default permissions to the GITHUB_TOKEN

Answer: A, B

Explanation:

Some actions may require a GITHUB_TOKEN as an input to authenticate and perform specific tasks, such as creating issues, commenting on pull requests, or interacting with the GitHub API. In such cases, you would need to explicitly pass the token to the action.

When making an authenticated GitHub API request, the GITHUB_TOKEN is required to authenticate the

request. This token is automatically provided by GitHub in the workflow, and it must be explicitly used when interacting with the GitHub API.

Question: 42

What will the output be for the following event trigger block in a workflow?

```
on:
  issues:
    types: [opened, edited]
  issue_comment:
    types:
      - created
```

A. It throws a workflow syntax error, pointing to the types definition in issue_comment event.
B. It throws a workflow syntax error, pointing to the types definition in issues event.
C. It runs the workflow when an issue is edited or when an issue comment created.
D. It runs the workflow when an issue or issue comment in the workflow's repository is created or modified.
E. It runs the workflow when an issue is created or edited, or when an issue or pull request comment is created.

Answer: C

Explanation:

The provided event trigger block specifies two types of events:
For issues: the workflow triggers on opened or edited issues.
For issue_comment: the workflow triggers when an issue comment is created.
This configuration ensures the workflow will run when either an issue is opened or edited, or an issue comment is created.

Question: 43

A 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: A

Explanation:

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.

Question: 44

Which default GitHub environment variable indicates the owner and repository name?

A. REPOSITORY NAME
B. GITHUB REPOSITORY
C. ENV REPOSITORY
D. GITHUB WORKFLOW REPO

Answer: A

Explanation:

The GITHUB_REPOSITORY environment variable contains the owner and repository name in the format owner/repository. It is automatically provided by GitHub Actions and can be used to reference the repository in workflows.

Question: 45

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

B. Add the check_run webhook event as a trigger for the workflow when the code quality integration is completed
C. Add the pull_request webhook event as a trigger for the workflow when the code quality integration is synchronized
D. Add the deployment webhook event as a trigger for the workflow when the code quality integration is completed

Answer: B

Explanation:

The check_run event is triggered when a check (such as a code quality check) completes, including when the status of a check changes. By adding this event as a trigger, you can initiate a follow-up workflow when the code quality integration finishes its checks.

Question: 46

What metadata file in a custom action defines the main entry point?

A. action.js
B. index.js
C. action.yml
D. main.yml

Answer: C

Explanation:

The action.yml file is the metadata file in a custom GitHub Action that defines the main entry point, including information such as the inputs, outputs, description, and the runs key that specifies the main entry point (e.g., a script or a Docker container).

Question: 47

Which workflow command would output the debug message "action successfully debugged"

A. echo :debug::message=action successfully debugged"
B. echo "debug-action successfully debugged"
C. echo "::debug::action successfully debugged"
D. echo ":debug:action successfully debugged:"

Answer: C

Explanation:

The ::debug:: syntax is used to output debug messages in GitHub Actions workflows. This command will print the message "action successfully debugged" in the debug logs when the workflow runs.

Question: 48

Which workflow event is used to manually trigger a workflow run?

A. create
B. workflow_dispatch
C. workflow_run
D. status

Answer: B

Explanation:

The workflow_dispatch event is used to manually trigger a workflow run in GitHub Actions. You can specify this event in the workflow file to allow users to manually trigger the workflow from the GitHub UI, often with optional input parameters.

Question: 49

Which of the following statements are true regarding the use of GitHub Actions on a GitHub Enterprise Server instance? (Choose three.)

A. Use of GitHub Actions on GitHub Enterprise Server requires a persistent internet connection
B. Actions created by GitHub are automatically available and cannot be disabled
C. Most GitHub authored actions are automatically bundled for use on GitHub Enterprise Server
D. Third party actions can be used on GitHub Enterprise Server by configuring GitHub Connect
E. Actions must be defined in the .github repository
F. Third party actions can be manually synchronized for use on GitHub Enterprise Server

Answer: A, D, F

Explanation:

GitHub Actions on GitHub Enterprise Server often requires an internet connection, especially for accessing actions from the GitHub Marketplace or third-party actions unless they are manually synced to the server. To use third-party actions on GitHub Enterprise Server, GitHub Connect can be used to establish a connection between the server and GitHub.com, enabling access to third-party actions.
Third-party actions can also be manually synchronized to the GitHub Enterprise Server, making them available for use in workflows.

Question: 50

You need to trigger a workflow using the GitHub API for activity that happens outside of GitHub. Which workflow event do you use?

A. check_suite
B. workflow_run
C. deployment
D. repository_dispatch

Answer: D

Explanation:

The repository_dispatch event allows you to trigger a workflow in response to external activity. It is commonly used when you need to trigger a workflow from outside GitHub, such as from another system or service, by sending a request to the GitHub API. This event provides flexibility to integrate with various external systems and trigger workflows in a GitHub repository.

Question: 51

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 a duplicate entry for SuperSecret in the encrypted secret store and specify MyRepo as the scope.
B. Create MyRepo_SuperSecret in GitHub encrypted secrets to specify the scope to MyRepo.
C. Create a file with the SuperSecret. information in the .qithub/secrets folder in MyRepo.
D. Create and access SuperSecret from the secrets store in MyRepo.

Answer: B

Explanation:

To scope a secret to a specific repository, you can create a new secret with a name like MyRepo_SuperSecret in the secrets section of the MyRepo repository's settings. This ensures that the secret is specific to that repository and can be used within its workflows.

Question: 52

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

Answer: A, D, F

Explanation:

You can define environment variables for the entire workflow by using the env key at the top level of the workflow file. These environment variables will be available to all jobs and steps within the workflow.
Environment variables can also be set within the run attribute of a job step, and these variables will be scoped only to that specific step.
You can set environment variables for specific steps within a job by using jobs.<job_id>.steps[*].env, which allows you to define variables that will only be available to that step.

Question: 53

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

Explanation:

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.

Question: 54

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

Explanation:

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.

Question: 55

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: C

Explanation:

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.

Question: 56

Which default environment variable specifies the branch or tag that triggered a workflow?

A. GITHUB_TAG
B. GITHUB_REF
C. ENV_BRANCH
D. GITHUB_BRANCH

Answer: B

Explanation:

The GITHUB_REF environment variable specifies the branch or tag that triggered the workflow. It contains the full reference to the branch or tag, such as refs/heads/main for a branch or refs/tags/v1.0 for a tag.

Question: 57

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

Explanation:

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.

Question: 58

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

Explanation:

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.

Question: 59

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

Explanation:

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.

Question: 60

Which syntax correctly accesses a job output (output1) of an upstream job (job1) from a dependent job within a workflow?

A. ${{needs.job1.outputs.output1}}
B. ${{needs.job1.output1}}
C. ${{depends.job1.output1}}
D. ${{job1.outputs.output1}}

Answer: A

Explanation:

The needs context is used to reference the outputs of jobs that are dependencies of the current job. In this case, needs.job1.outputs.output1 correctly accesses the output of output1 from the job job1 in the dependent job.

Question: 61

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

Explanation:

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.

Question: 62

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 job runs-on must specify a Linux machine with Docker installed.
C. The referenced action must be hosted on Docker Hub.
D. The action must be published to the GitHub Marketplace.

Answer: B

Explanation:

For a Docker container action to run in a GitHub Actions workflow, the runner must have Docker installed. The runs-on attribute of the job should specify an environment that supports Docker, typically a Linux environment (e.g., ubuntu-latest), since Docker is widely supported and commonly used in Linux-based environments.

Question: 63

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

Explanation:

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.

Question: 64

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

Explanation:

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.

Question: 65

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 permissions

Answer: C

Explanation:

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.

Question: 66

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

Explanation:

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.

Question: 67

Which run: command will set a step's output?

A. run: echo "MY_OUTPUT=foo" >> $GITHUB_OUTPUT
B. run: export MY_OUTPUT=foo
C. run: echo ${{ $GITHUB_OUTPUT=foo }}
D. run: echo "::set-env name=MY OUTPUT::foo"

Answer: A

Explanation:

The $GITHUB_OUTPUT file is used to pass data from one step to another in GitHub Actions. The echo command appends the key-value pair to this file, which sets the output variable (MY_OUTPUT) for the current step.

Question: 68

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

A. The action's repository includes @actions/core in the root directory.
B. The action's repository name includes the keyword "Docker."
C. The action.yml metadata file references a Dockerfile file.
D. The action.yml metadata file has the runs.using value set to Docker.

Answer: D

Explanation:

In a Docker container action, the action.yml file includes the runs.using field, which is set to docker to specify that the action runs inside a Docker container. This is the key indicator that the action is a Docker container action.

Question: 69

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

Explanation:

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).

Question: 70

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

Explanation:

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.

Question: 71

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

Explanation:

When troubleshooting a self-hosted runner, you can access the _diag folder located in the selfhosted runner's installation directory. This folder contains diagnostic logs that can help you identify the root cause of issues, such as connection problems.

Question: 72

As a developer, you need to create a custom action written in Python. Which action type should you choose?
As a developer, you need to create a custom action written in Python. Which action type should you choose?

A. JavaScript action
B. composite run step
C. Python action
D. Docker container action

Answer: D

Explanation:

A Docker container action is ideal for custom actions that require specific environments or dependencies, such as Python. By creating a Docker container, you can define the environment with the necessary Python version and dependencies, and your Python code can run inside that container.