Functions in Workflows

GitHub Actions offers a set of built-in functions that you can use in expressions. These functions allow you to perform various operations, including comparisons, string manipulations, and data transformations.

Data Type Conversions

Before diving into the functions, it's essential to understand how GitHub Actions converts data types when working with functions:

- **Null**: Converts to an empty string `''`.
- **Boolean**: Converts to `'true'` or `'false'`.
- **Number**: Converts to decimal format or exponential notation for large numbers.
- **Array**: Not converted to a string.
- **Object**: Not converted to a string.

Commonly Used Functions

- `contains`: checks if a value contains another value. It returns `true` if the search value is found in the target value.
- `fromJSON`: converts a value into a JSON object or data type. It's used to work with JSON data in expressions or convert environment variables from strings to JSON.
- `startsWith`: checks if a string starts with a specified substring. It returns true if the string starts with the given substring.
- `hashFiles`: used to generate a hash based on a single or multiple files. One use-case is to generate caching keys based on dependency lock files.

For a full list of available functions, check GitHub Actions's functions page.

Status Check Functions

These functions are used in if conditionals to determine the success or failure of previous steps or jobs.

- `success()`: returns `true` when none of the previous steps have failed or been canceled.
- `failure()`: returns `true` when any previous step of a job fails.
- `cancelled()`: returns `true` if the workflow was canceled.
- `always()`: always returns `true` and ensures that a step executes, even if previous steps failed or if the workflow was canceled.

Testing Functions in Workflows

1. Create Workflow File:

- Create a new file named `11-01-Using-Functions.yml` in the `.github/workflows` directory of your repository.

```yaml
name: 11-01-Using-Functions

on:
  pull_request:
  workflow_dispatch:

jobs:
  echo1:
    runs-on: ubuntu-latest
    steps:
      - name: Print PR title
        run: echo "${{ github.event.pull_request.title }}"
      - name: Print PR labels
        run: |
          cat << EOF
          ${{ toJSON(github.event.pull_request.labels) }}
          EOF
      - name: Bug step
        if: ${{ !cancelled() && contains(github.event.pull_request.title, 'fix') }}
        run: echo "I am a bug fix"
      - name: Sleep for 20 seconds
        run: sleep 20
      - name: Failing step
        run: exit 1
      - name: I will be skipped
        if: ${{ success() }}
        run: echo "I will print if previous steps succeed."
      - name: I will execute
        if: ${{ failure() }}
        run: echo "I will print if any previous step fails."
      - name: I will execute
        if: ${{ !cancelled() }}
        run: echo "I will always print, except when the workflow is cancelled."
      - name: I will execute when cancelled
        if: ${{ cancelled() }}
        run: echo "I will print if the workflow has been cancelled."
```

- Copy and paste the provided YAML configuration into this file.

2. Understanding the Workflow:

- This workflow is triggered both on pull requests (`pull_request`) and manually via the GitHub UI (`workflow_dispatch`).
- It defines a job (`echo1`) that runs on an Ubuntu latest runner.
- Within the job, various steps are defined to showcase the usage of GitHub Actions functions and expressions.

- These steps include printing the pull request title and labels, executing conditional steps based on the PR title, introducing delays, failing a step intentionally, and executing steps conditionally based on success, failure, or cancellation of previous steps.

3. Testing the Workflow:

- Commit and push the workflow file (`11-01-Using-Functions.yml`) to your repository.
- Navigate to the "Actions" tab in your GitHub repository.
- Manually trigger the workflow by clicking on the "Run workflow" button for the `11-01-Using-Functions` workflow or wait for it to be triggered by a pull request event.
- Once the workflow run completes, click on the job (`echo1`) to view its details.
- Review the logs of each step to observe the output and understand how each function and expression is evaluated.

4. Observing Output:

- Observe the output of each step to see the pull request title, labels, and the execution status of conditional steps based on various criteria.
- Pay attention to how conditional steps are executed based on the PR title and the outcome of previous steps.